Introduction to LArSoft

Erica Snider
Fermilab

LArSoft core support team:

Vito di Benedetto, Lynn Garren
Patrick Gartung, Gianluca Petrillo,
Saba Sehrish, Erica Snider

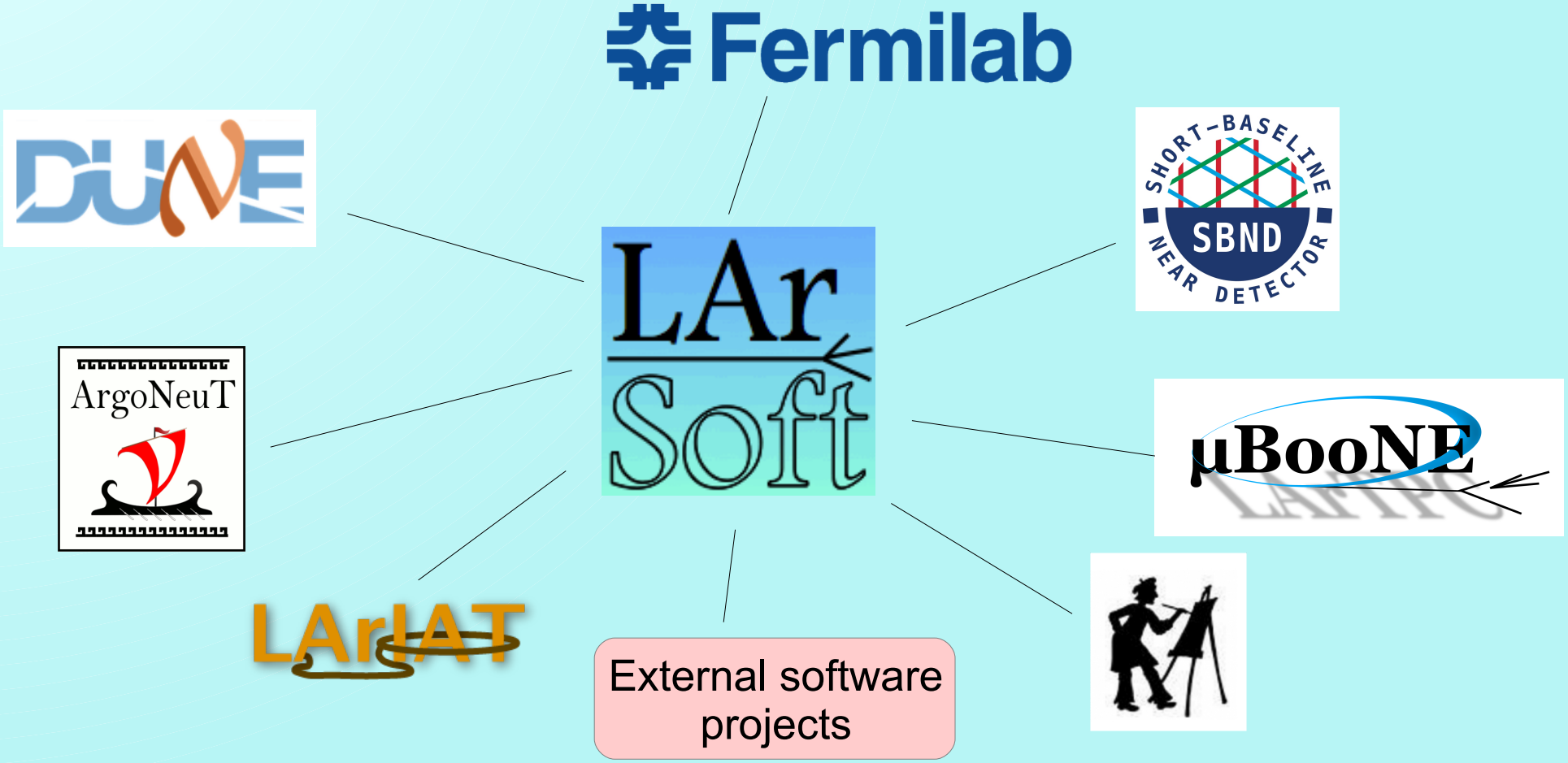
Young DUNE
September 16, 2016
Fermilab

Outline

- What is LArSoft?
- Operation of single-phase LAr TPC
- Reconstruction / data structure overview
  - Special algorithm notes
  - Additional reconstruction considerations
- Simulation overview
- LArSoft design principles
- Using LArSoft
- Resources

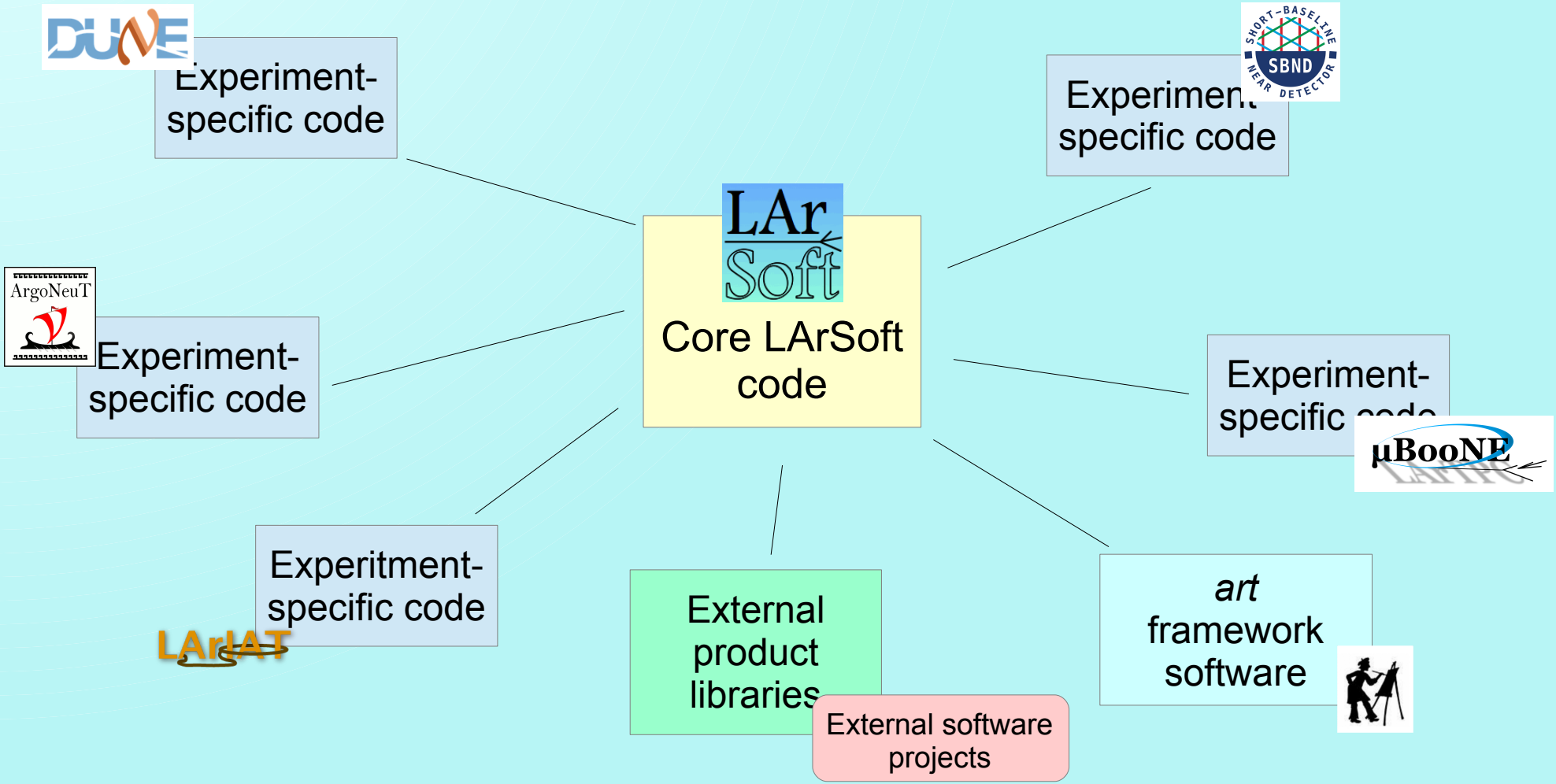
The slides today were updated from the 2015 art / LArSoft Course content.
See the course indico page for more information, material on LArSoft

What is LArSoft?: (1) **A collaboration** of experiments, Fermilab, other stakeholders



To provide integrated, experiment-independent software tools for LAr TPC neutrino experiments to perform simulation, reconstruction analysis.

What is LArSoft?: (2) A body of code that interfaces with experiment-specific, *art* framework and external product software

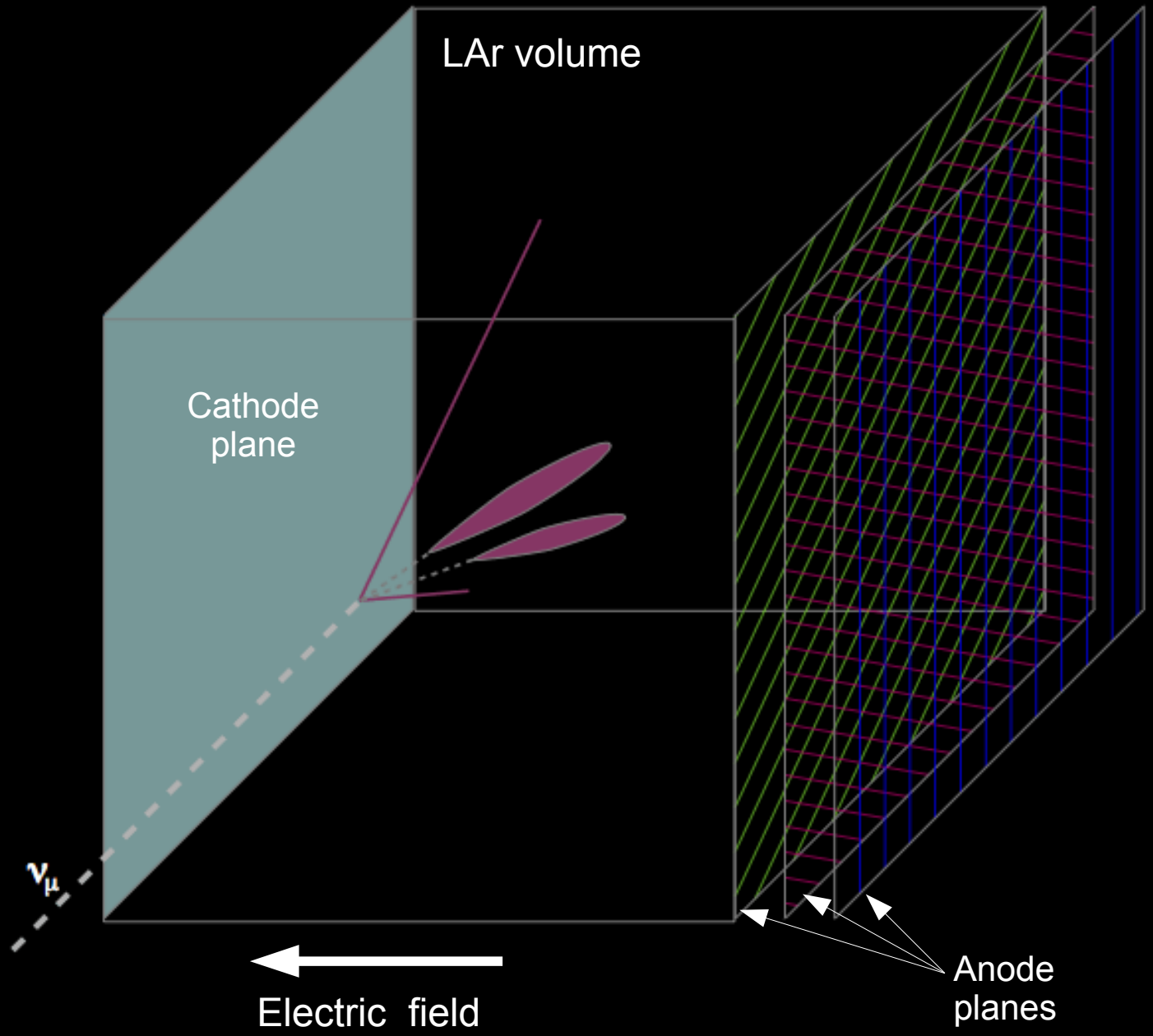


Experiments contribute common "core" LArSoft code

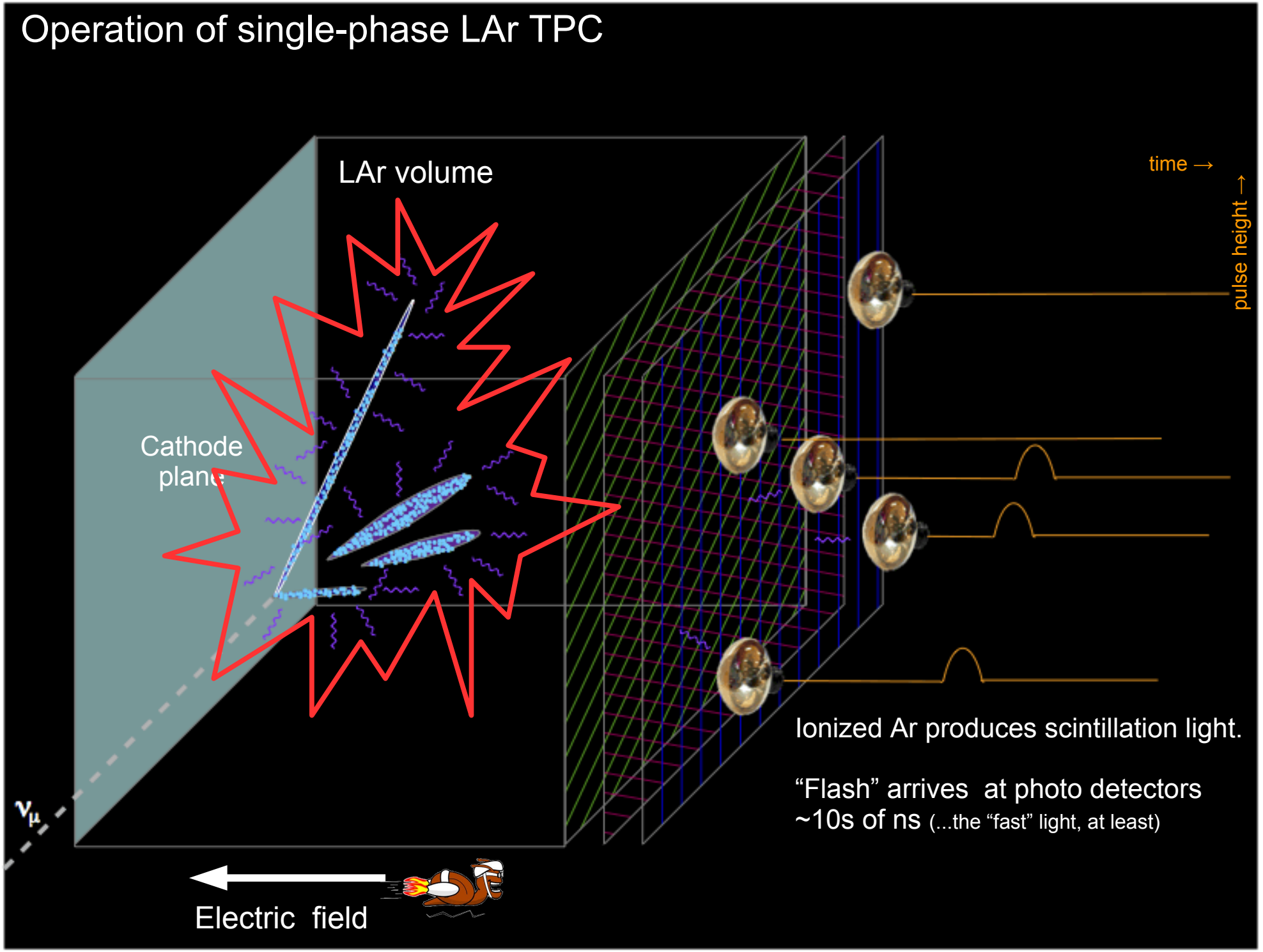
230k lines of C++ in core LArSoft
450k+ lines including expt code

Operation of a single-phase LAr TPC

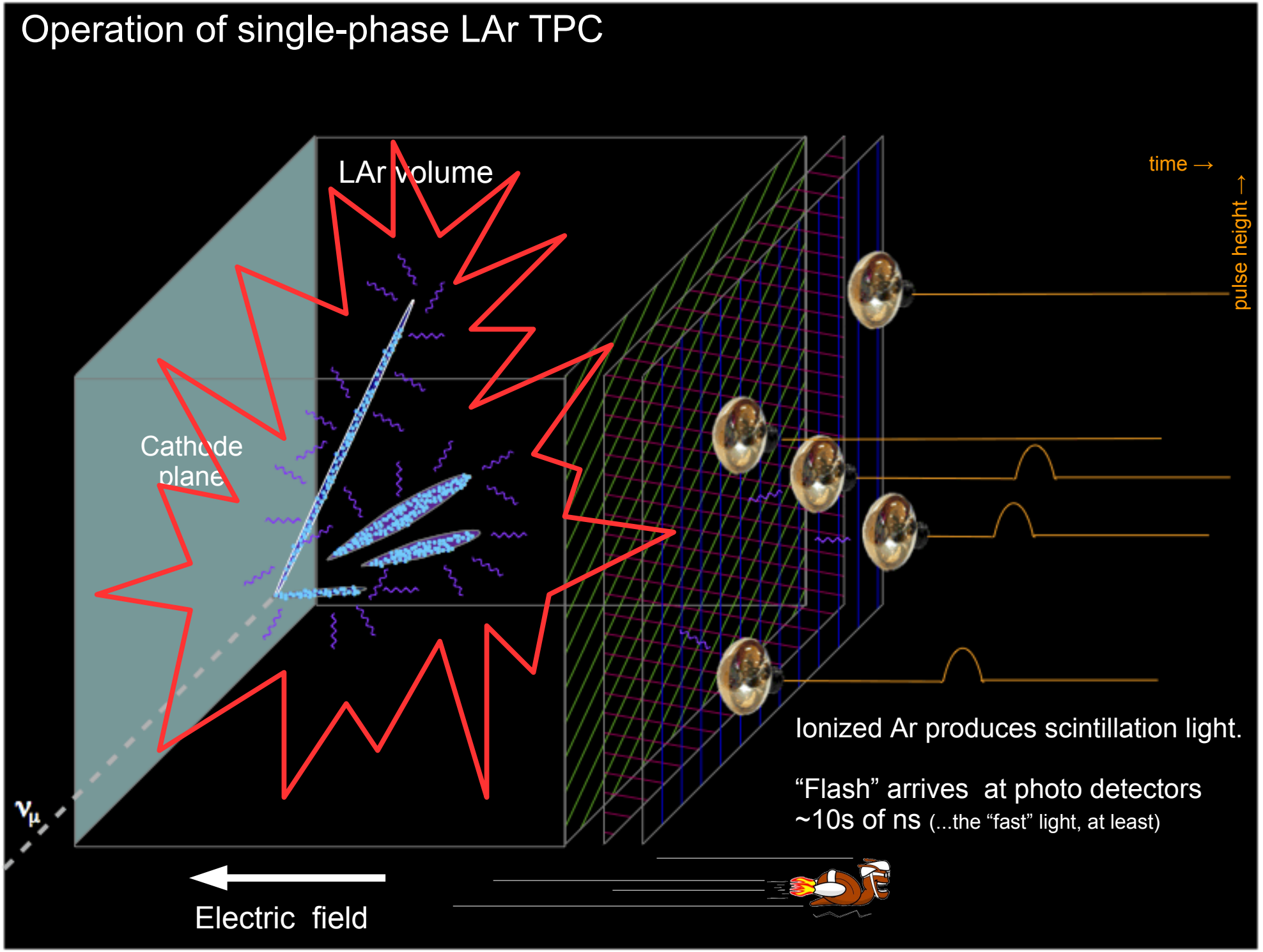
Operation of single-phase LAr TPC



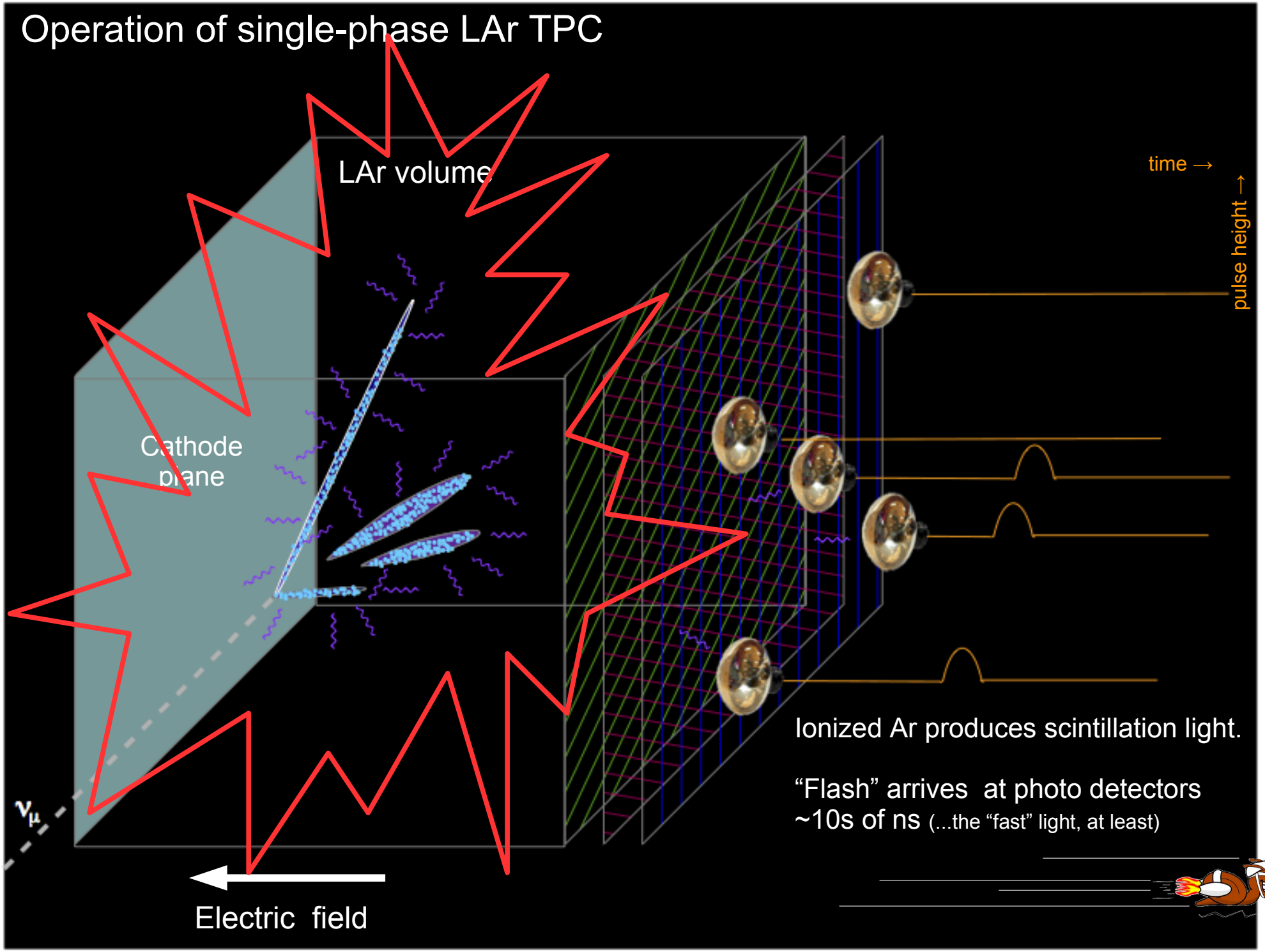
Operation of single-phase LAr TPC



Operation of single-phase LAr TPC



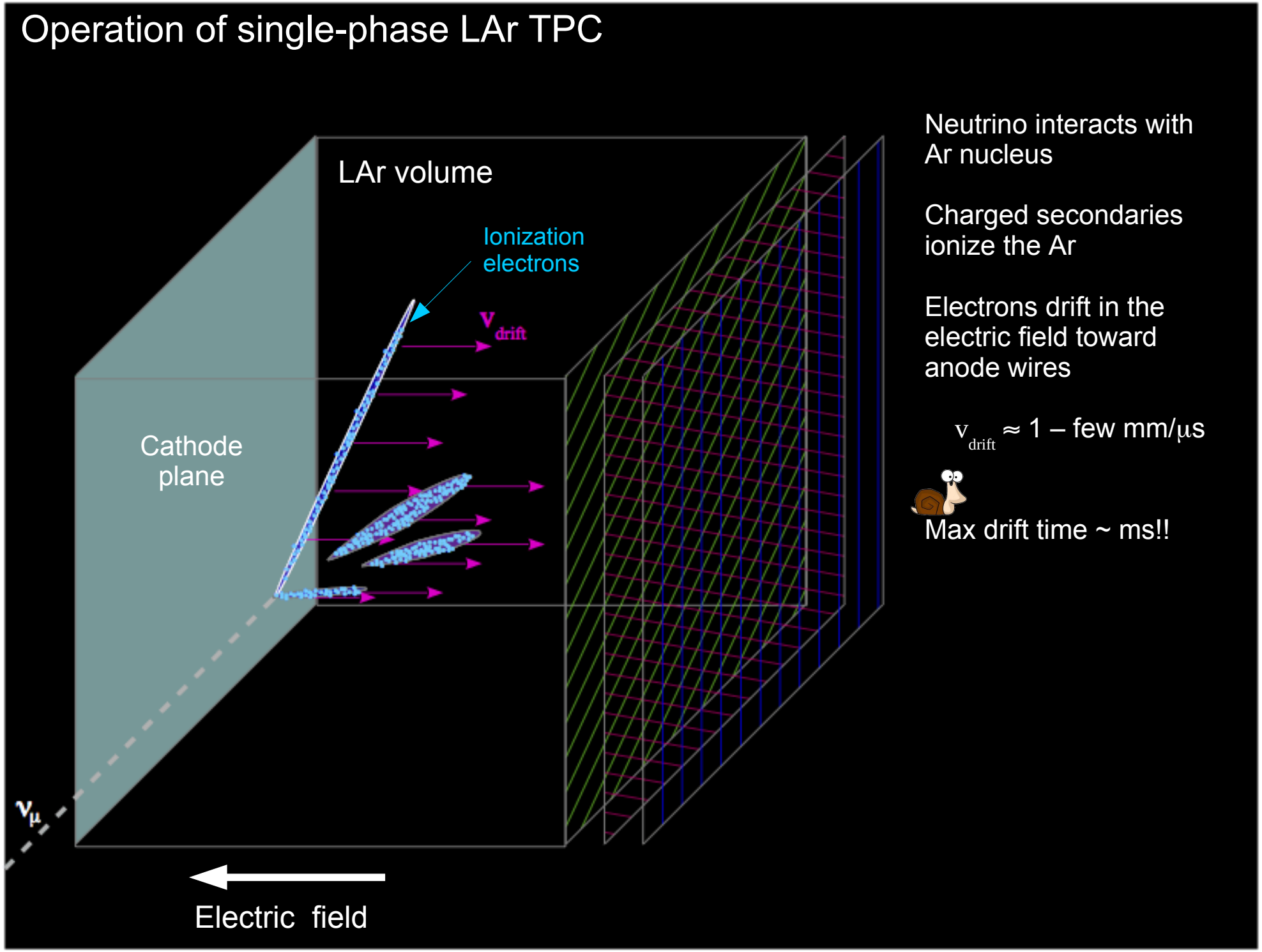
Operation of single-phase LAr TPC



Ionized Ar produces scintillation light.

"Flash" arrives at photo detectors ~10s of ns (...the "fast" light, at least)

Operation of single-phase LAr TPC



Neutrino interacts with Ar nucleus

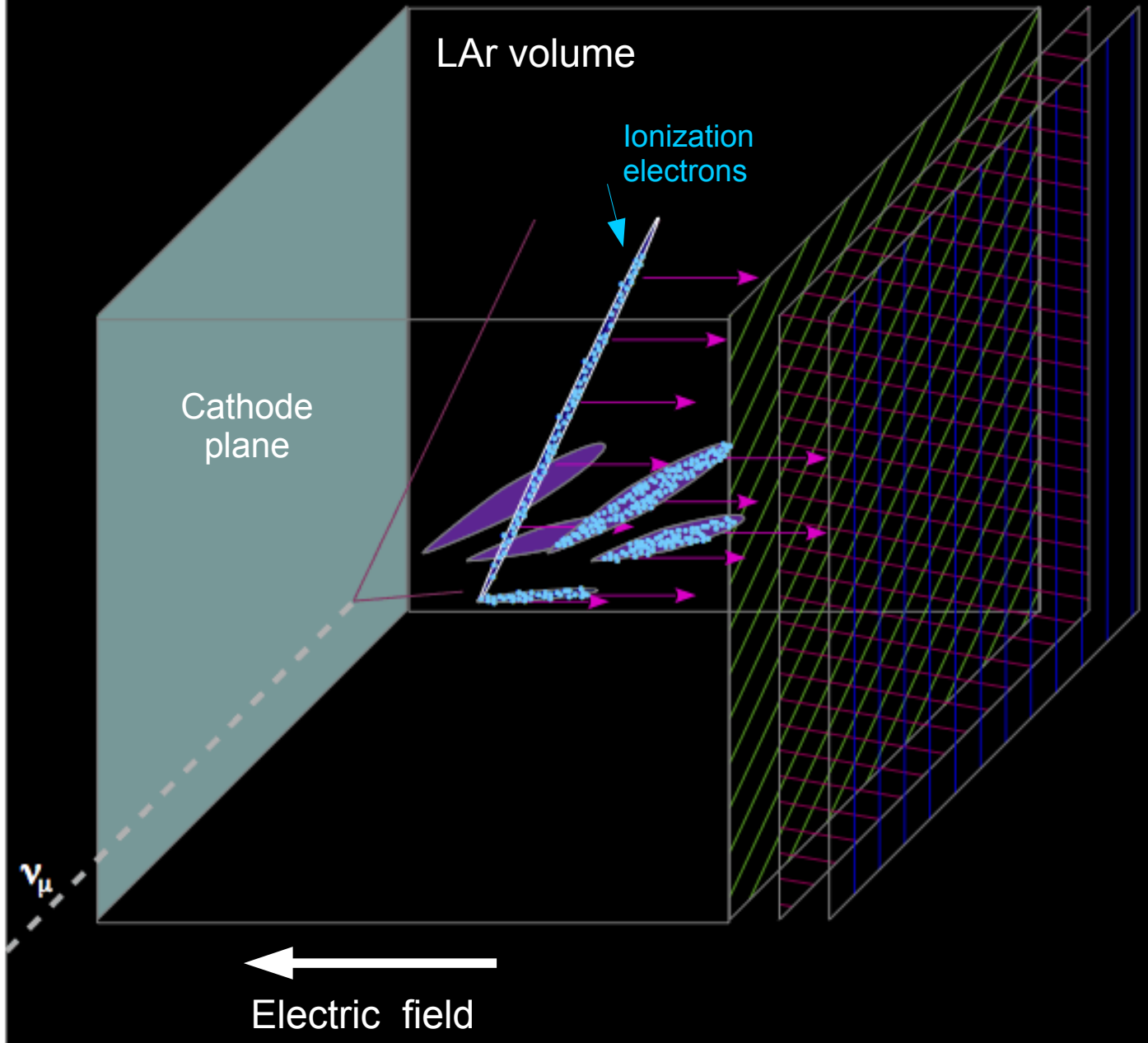
Charged secondaries ionize the Ar

Electrons drift in the electric field toward anode wires

$v_{drift} \approx 1$ – few mm/μs

Max drift time ~ ms!!

Operation of single-phase LAr TPC



Neutrino interacts with Ar nucleus

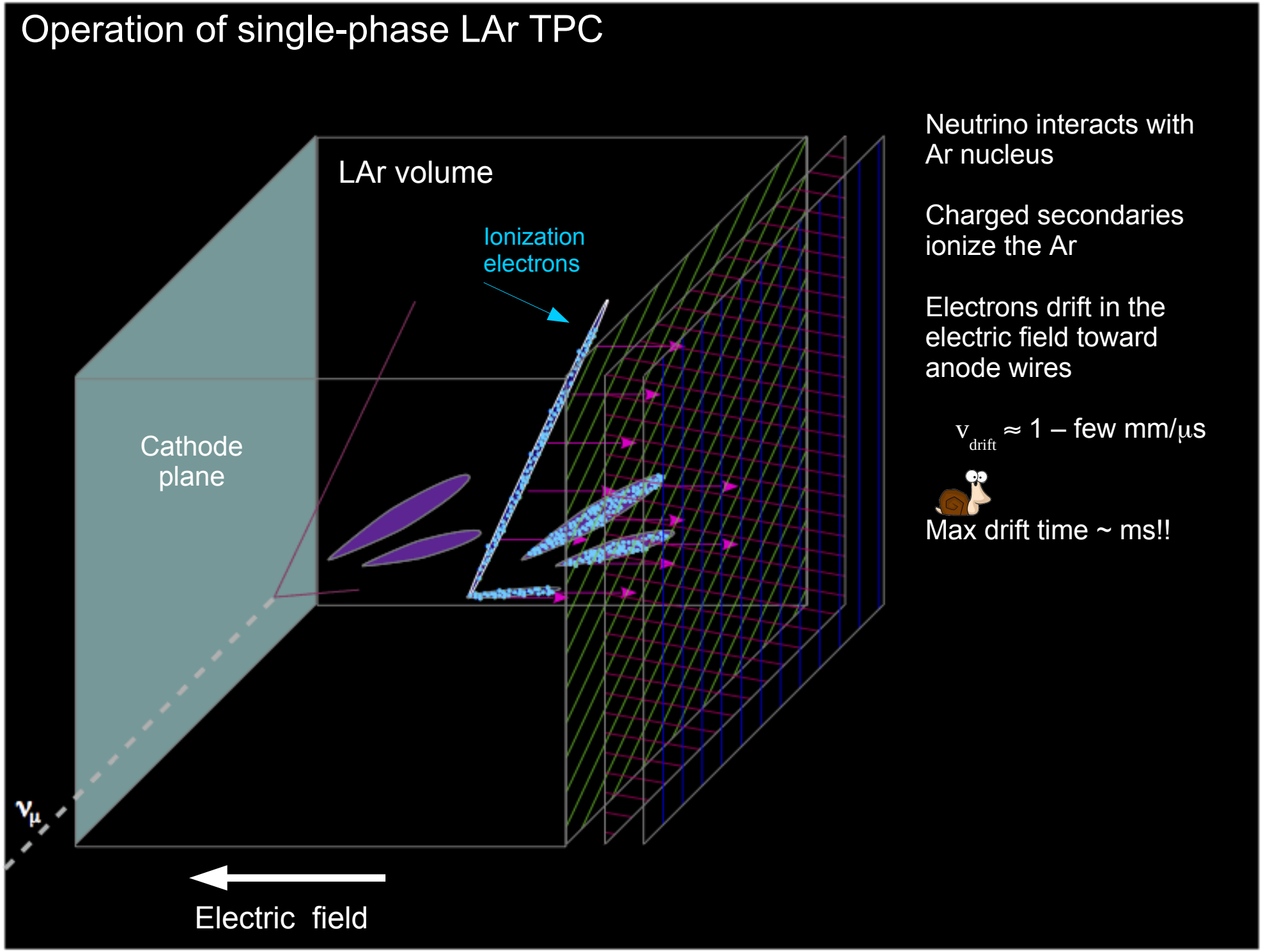
Charged secondaries ionize the Ar

Electrons drift in the electric field toward anode wires

$v_{drift} \approx 1$ – few mm/μs

Max drift time ~ ms!!

Operation of single-phase LAr TPC



Neutrino interacts with Ar nucleus

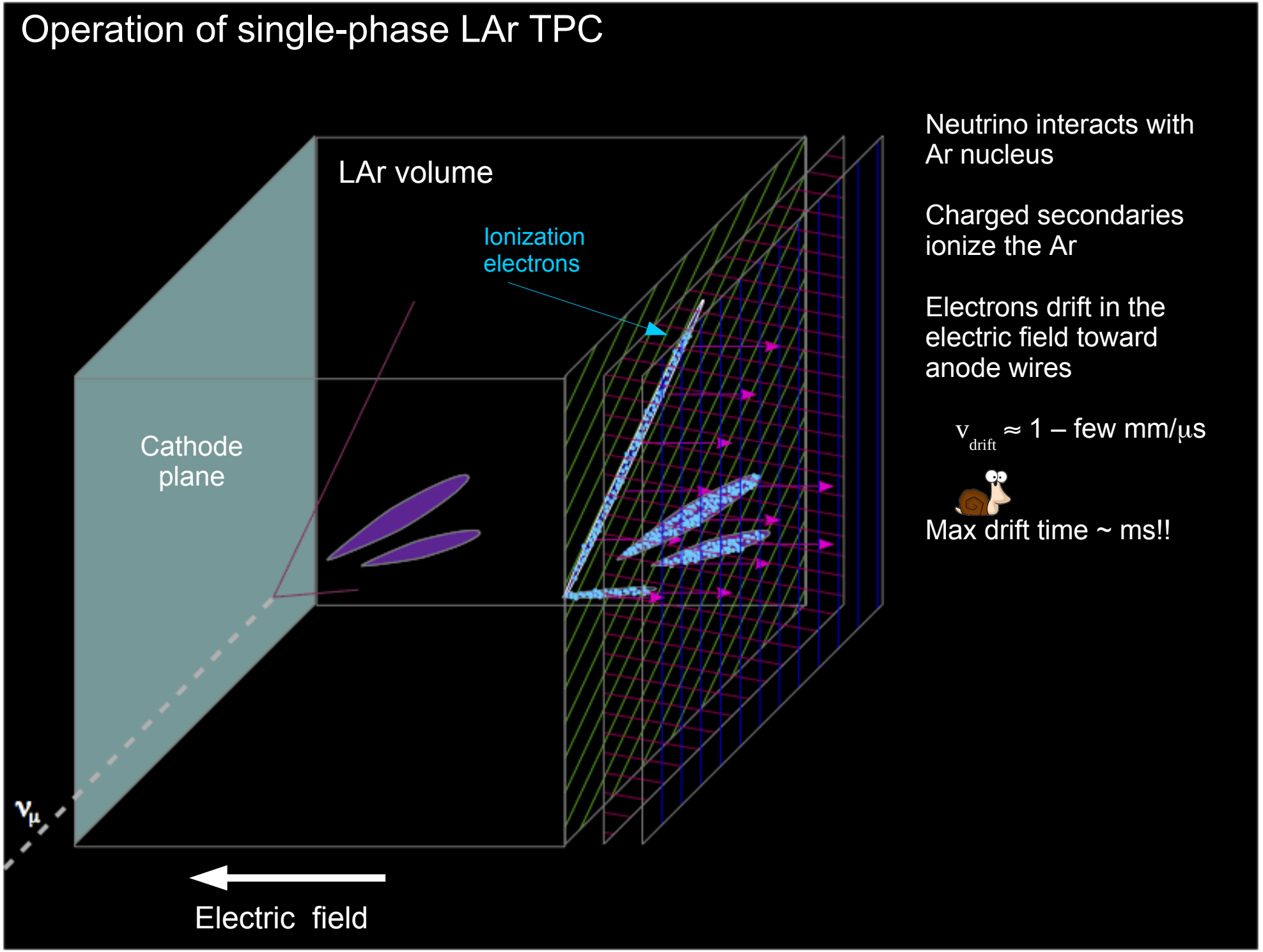
Charged secondaries ionize the Ar

Electrons drift in the electric field toward anode wires

$v_{drift} \approx 1$ – few mm/μs

Max drift time ~ ms!!

Operation of single-phase LAr TPC



Neutrino interacts with Ar nucleus

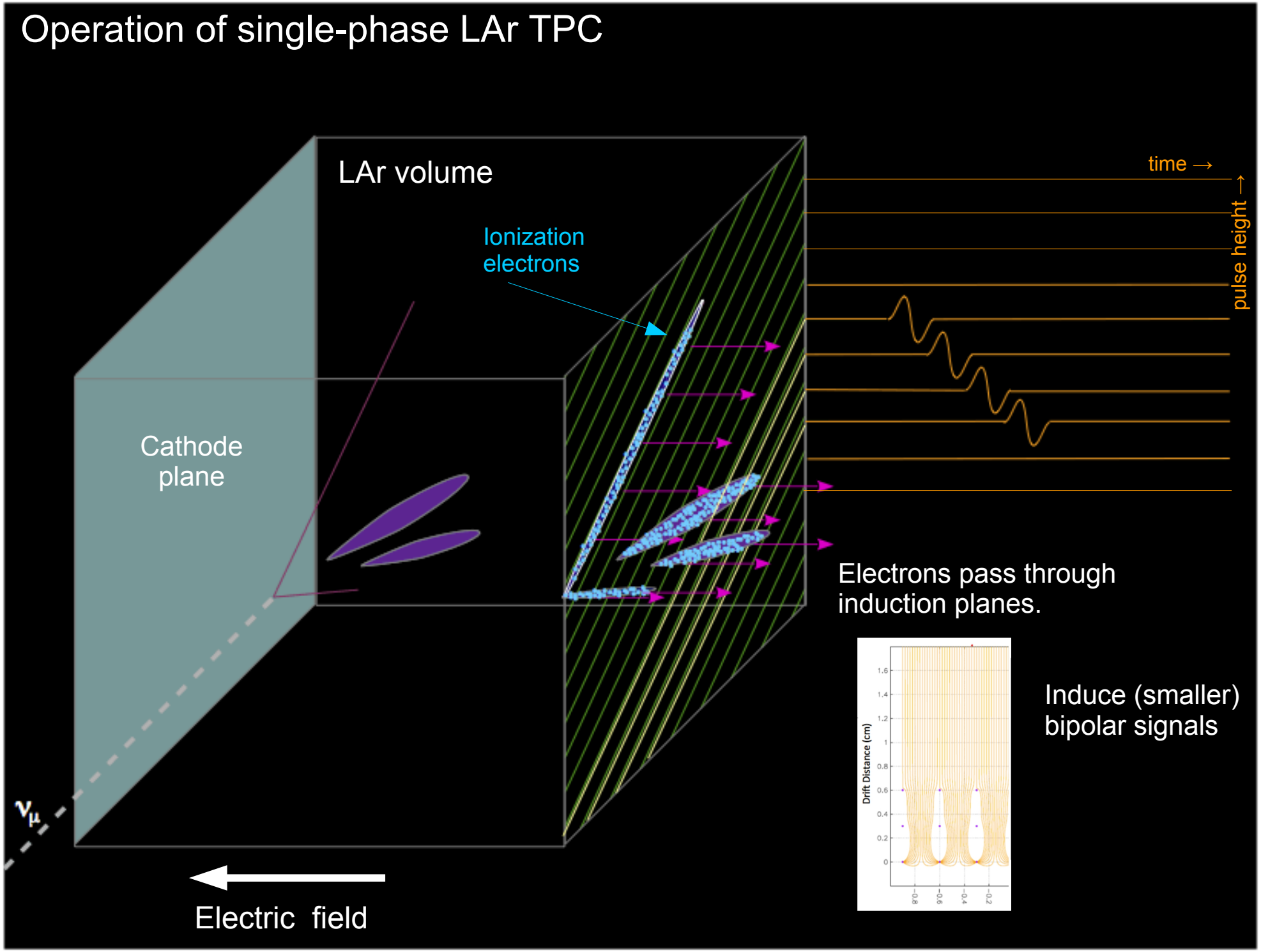
Charged secondaries ionize the Ar

Electrons drift in the electric field toward anode wires

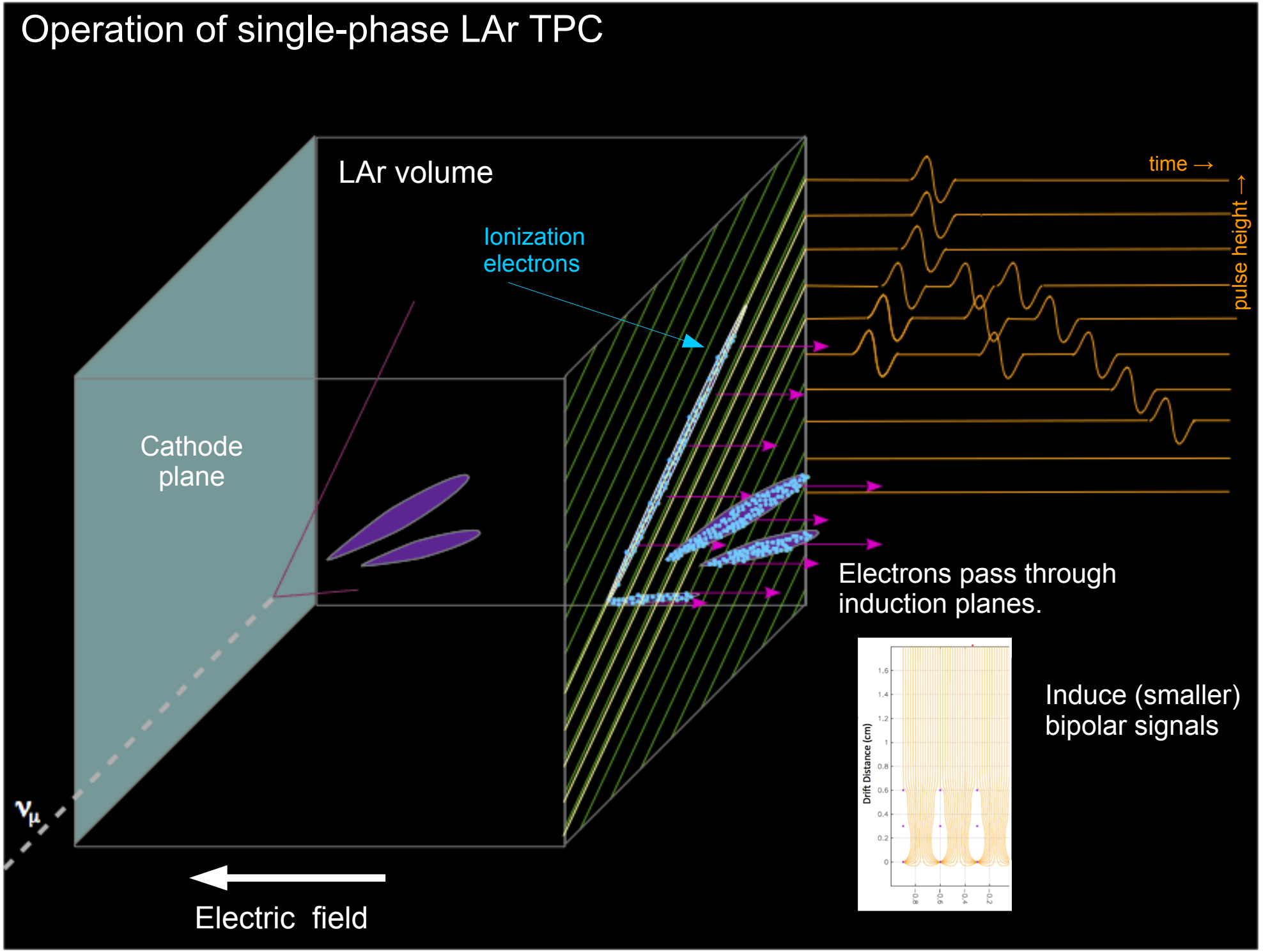
$v_{drift} \approx 1$ – few mm/μs

Max drift time ~ ms!!

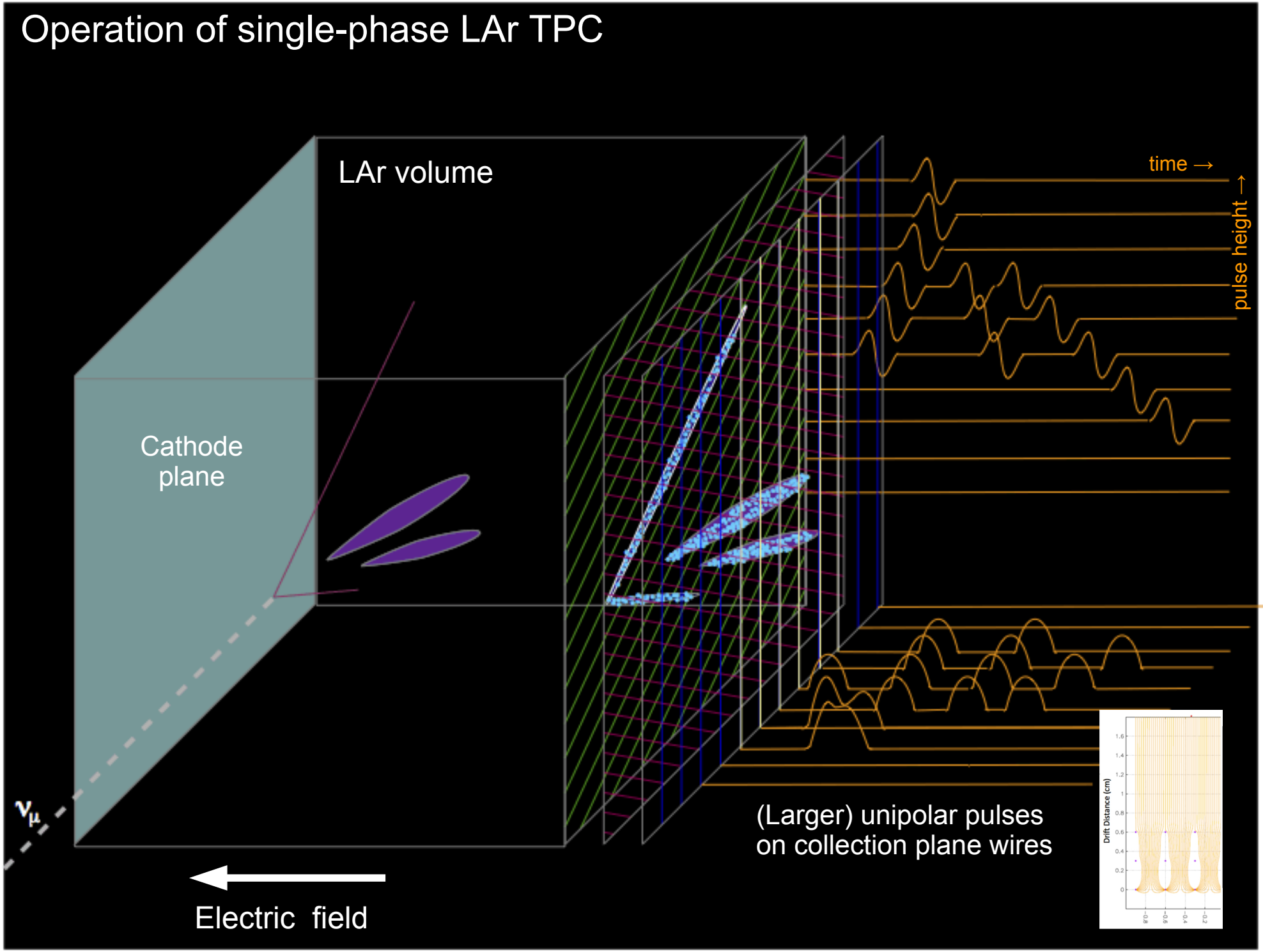
Operation of single-phase LAr TPC



Operation of single-phase LAr TPC



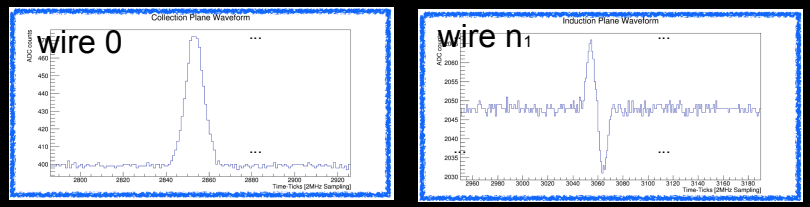
Operation of single-phase LAr TPC



Operation of single-phase LAr TPC

The job of the reconstruction:

To start with this...



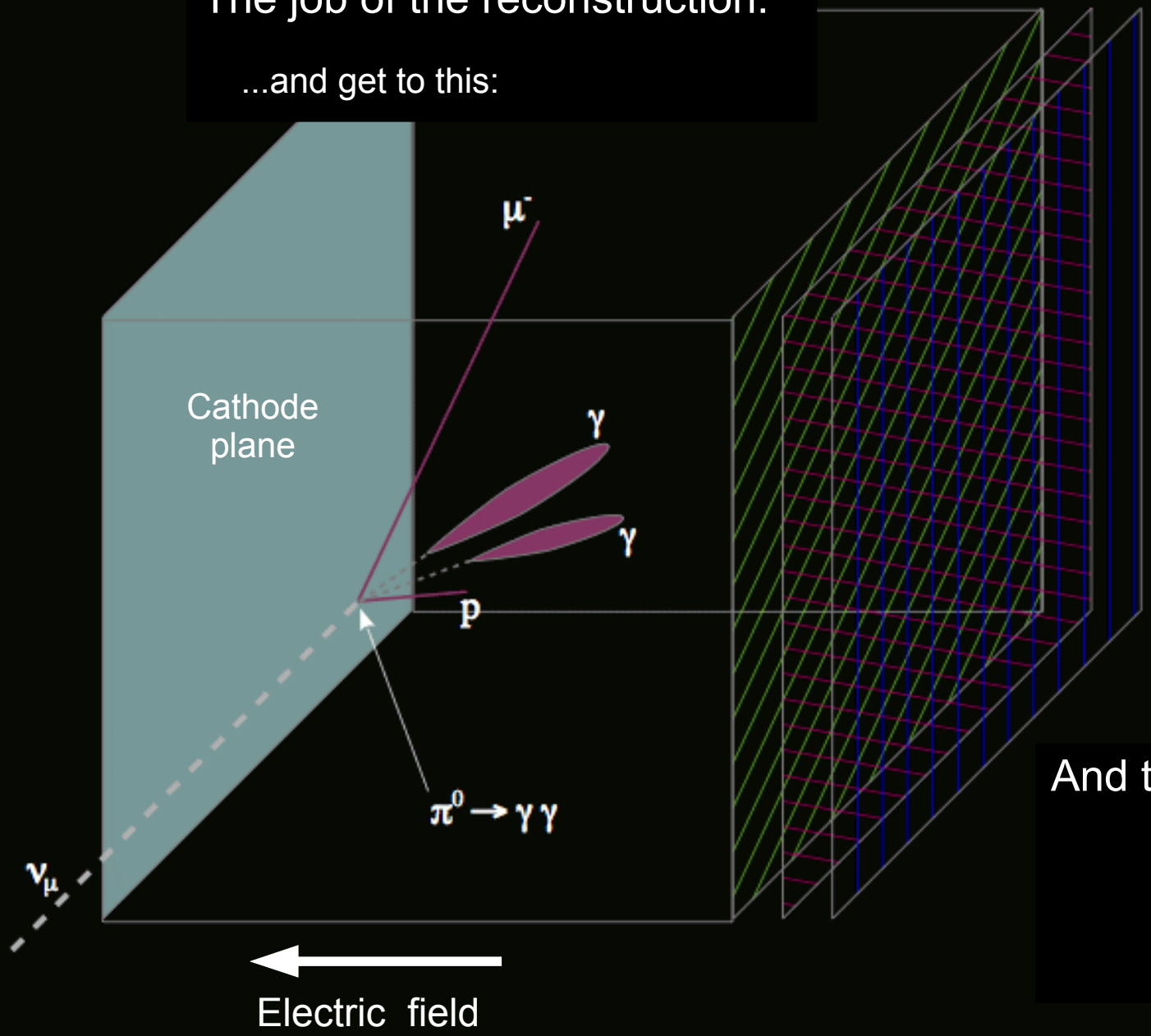
...

...

Operation of single-phase LAr TPC

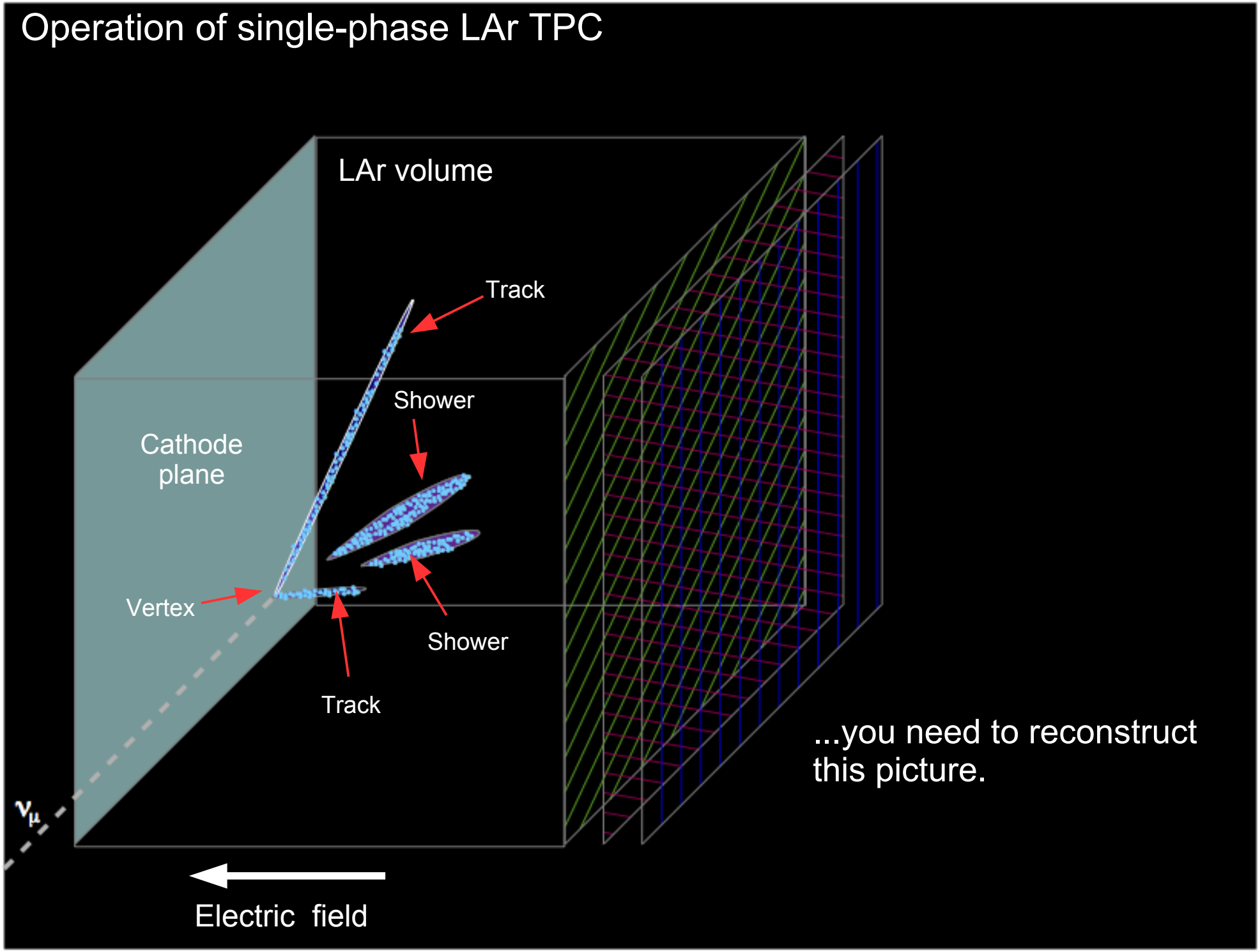
The job of the reconstruction:

...and get to this:

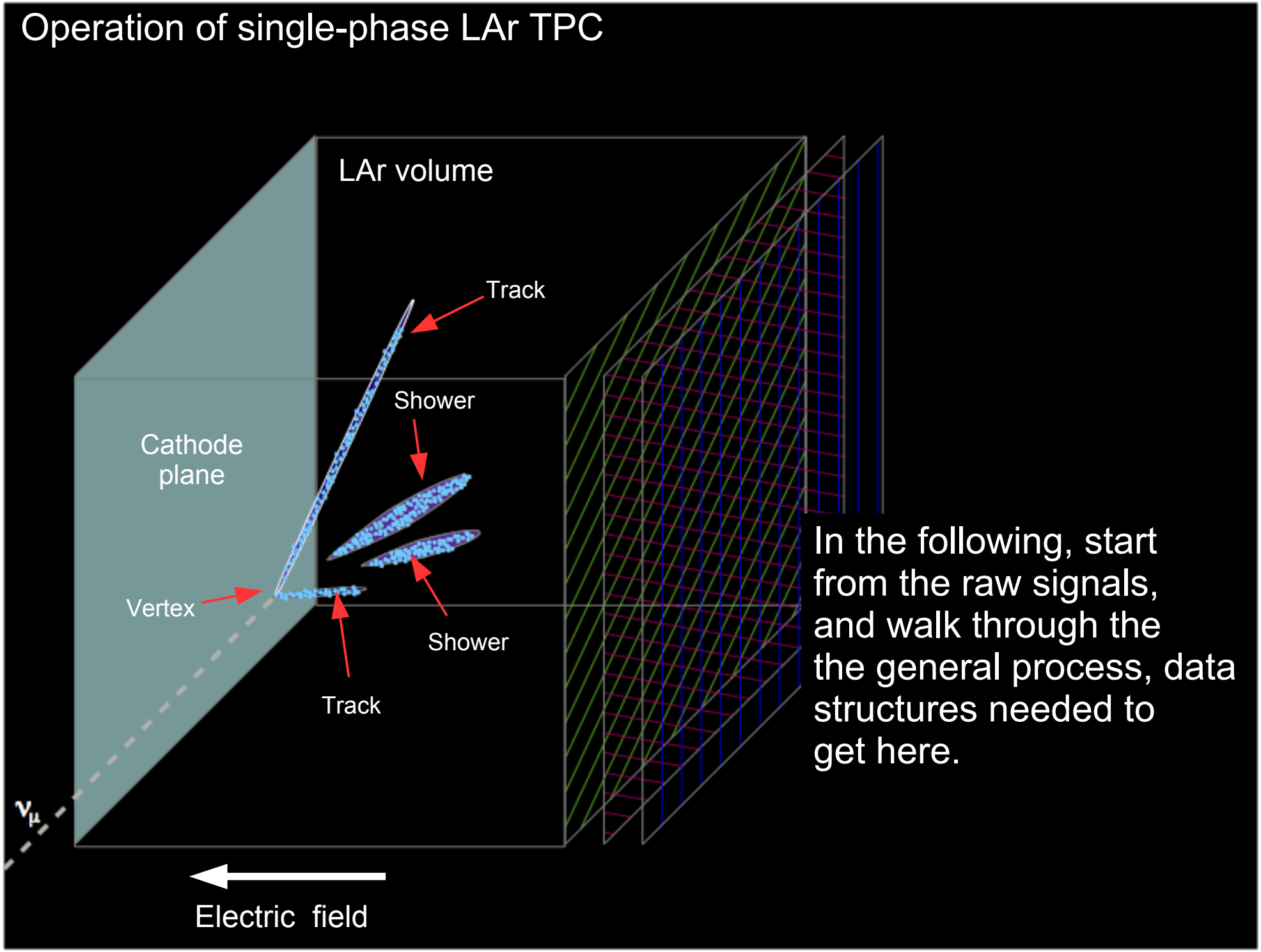


And to get here...

Operation of single-phase LAr TPC



Operation of single-phase LAr TPC



In the following, start from the raw signals, and walk through the the general process, data structures needed to get here.

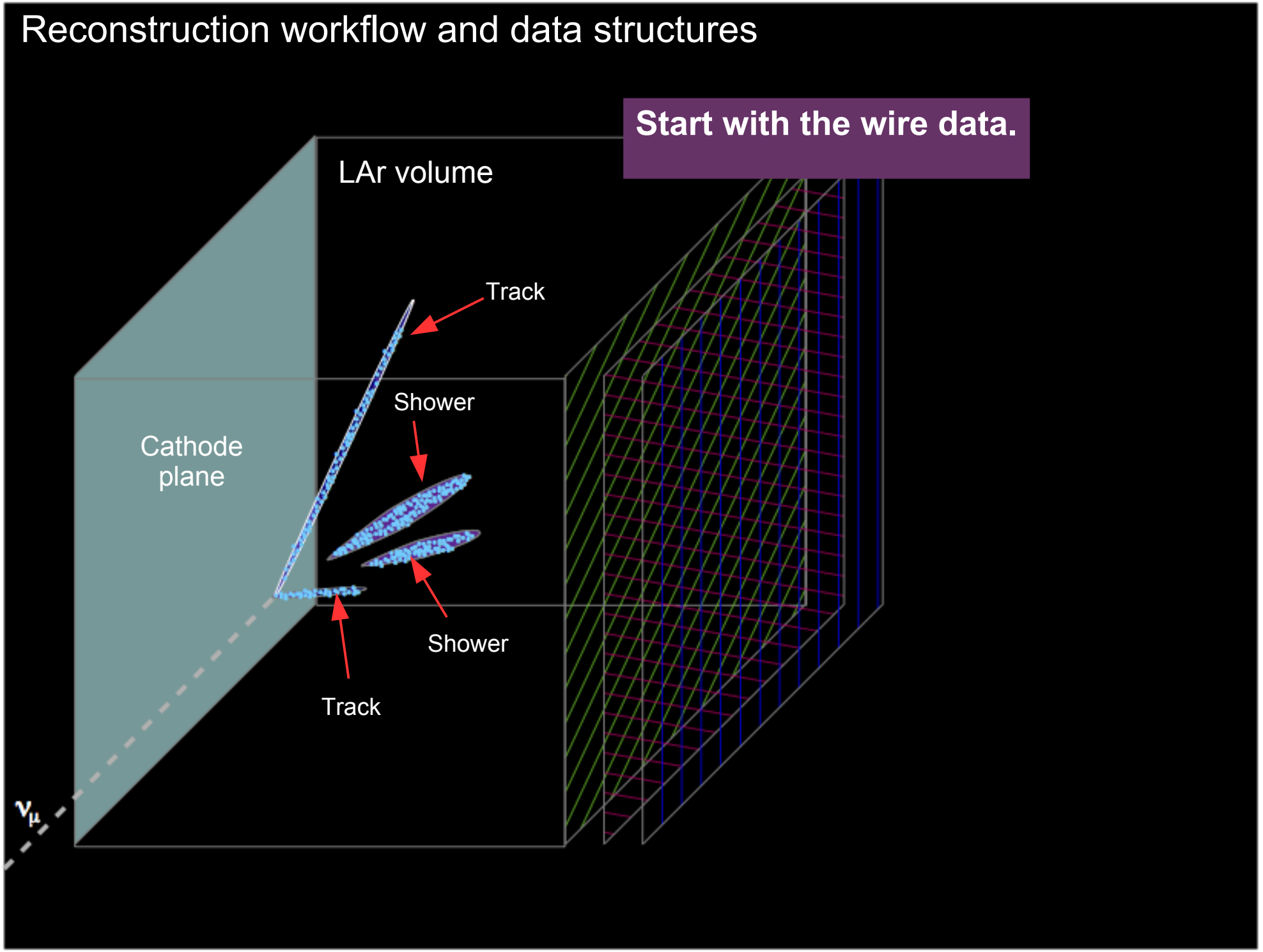
Reconstruction workflow and data structure overview

Comments on reconstruction workflow

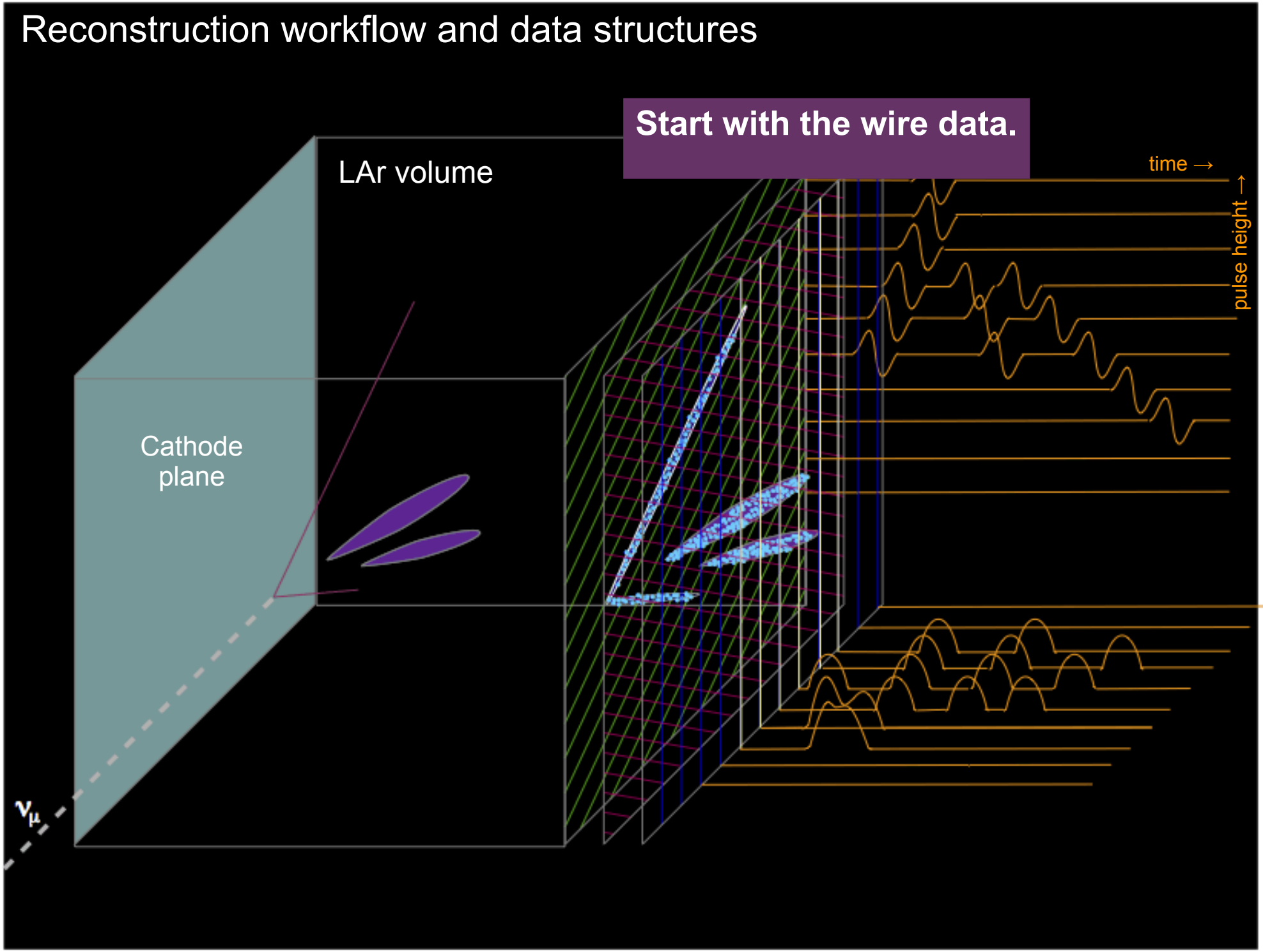
- What follows is a simple, highly idealized representation of the typical reconstruction workflow
- In practice, the reconstruction
  - is highly iterative
  - has multiple parallel algorithm workflow
  - has multiple steps interleaved

  as increasingly complex structures are extracted from the data
- Will come back to talk about this later

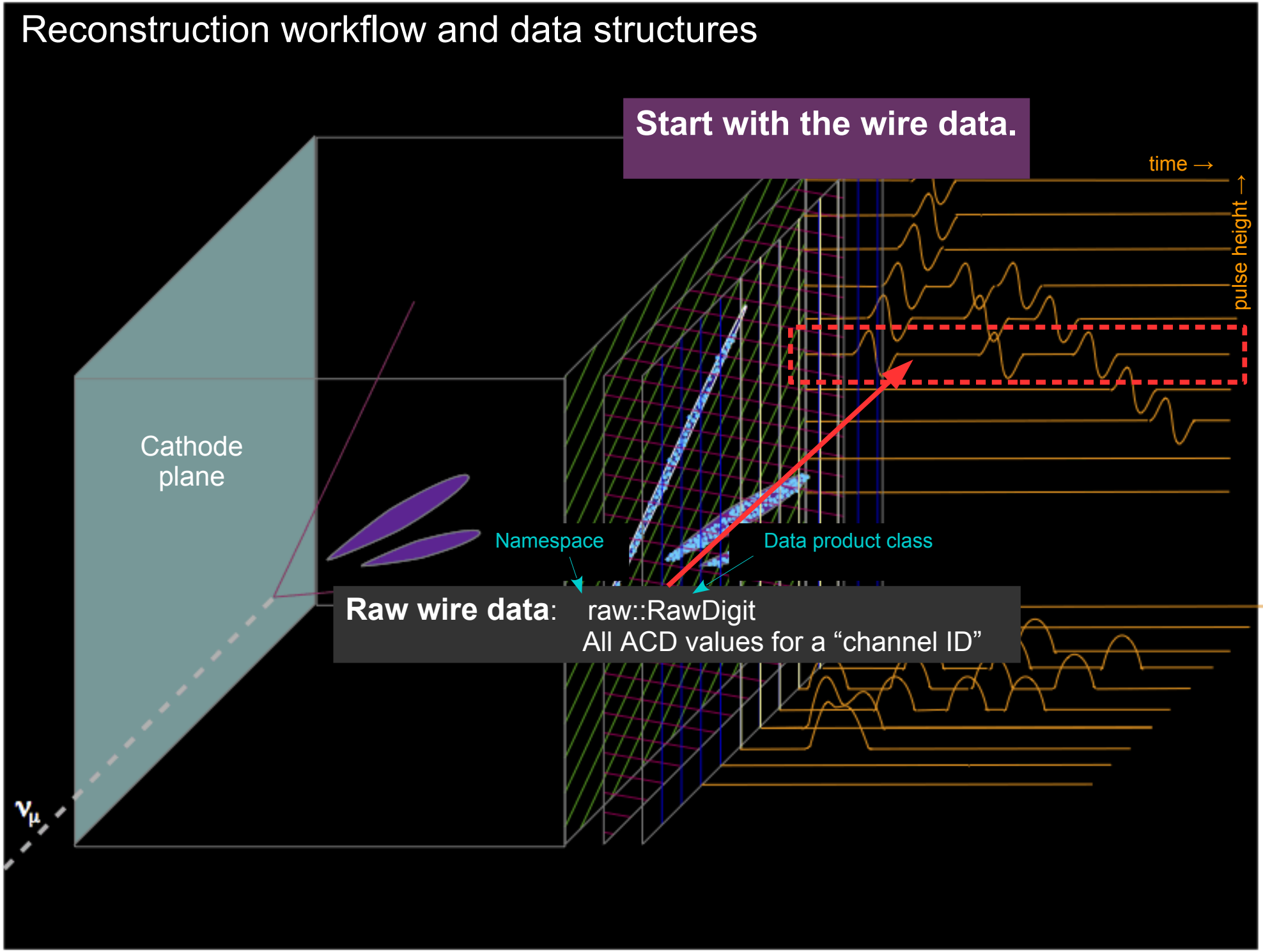
Reconstruction workflow and data structures



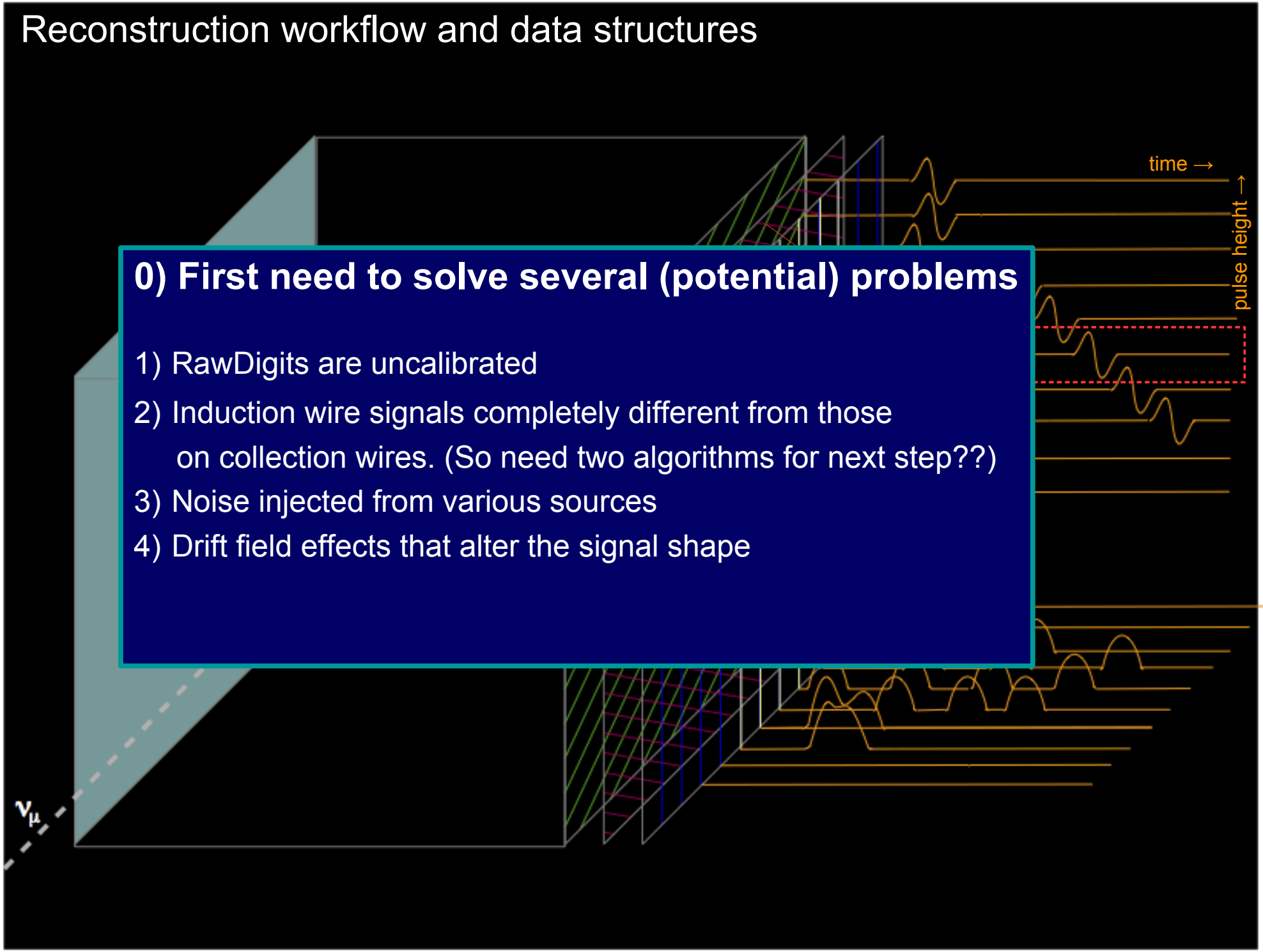
Reconstruction workflow and data structures



Reconstruction workflow and data structures



Reconstruction workflow and data structures



0) First need to solve several (potential) problems

1) RawDigits are uncalibrated
2) Induction wire signals completely different from those on collection wires. (So need two algorithms for next step??)
3) Noise injected from various sources
4) Drift field effects that alter the signal shape

(0) Signal processing and calibration

- Want to recover charge vs time on a wire
  - The level of detail may depend on the task
  - E.g., calorimetry may require more fidelity than pattern recognition
- Consider MicroBooNE example: fairly simple workflow at present
  - Calibration: baseline subtraction
  - Noise removal
  - Deconvolution
    - Removes effects of electronics response, field response

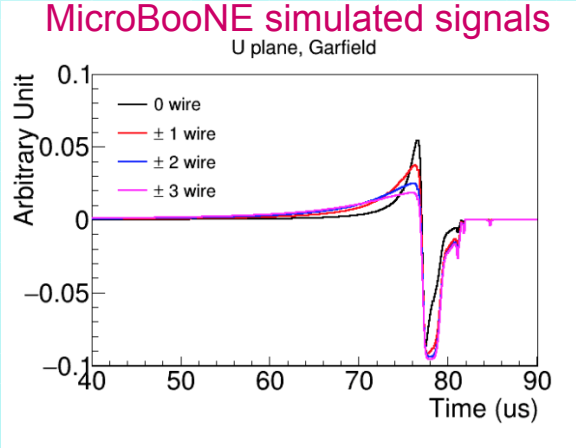
TPC signal formation

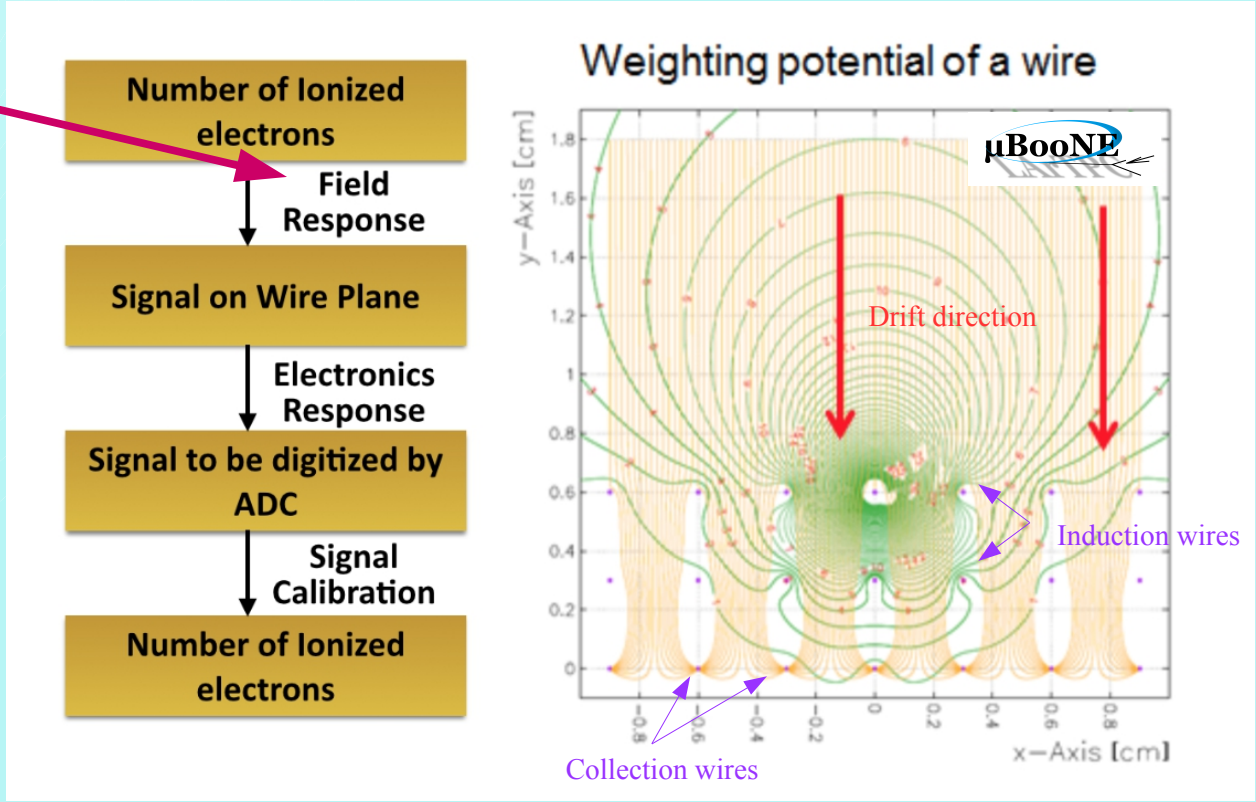
Induced current $i \propto q\,\vec{E}_w \cdot \vec{v}_q$

$\vec{E}_w$ = "weighting E field"

$\vec{v}_q$ = velocity of charge 'q'

In principle, see charge drifting to neighboring wires too





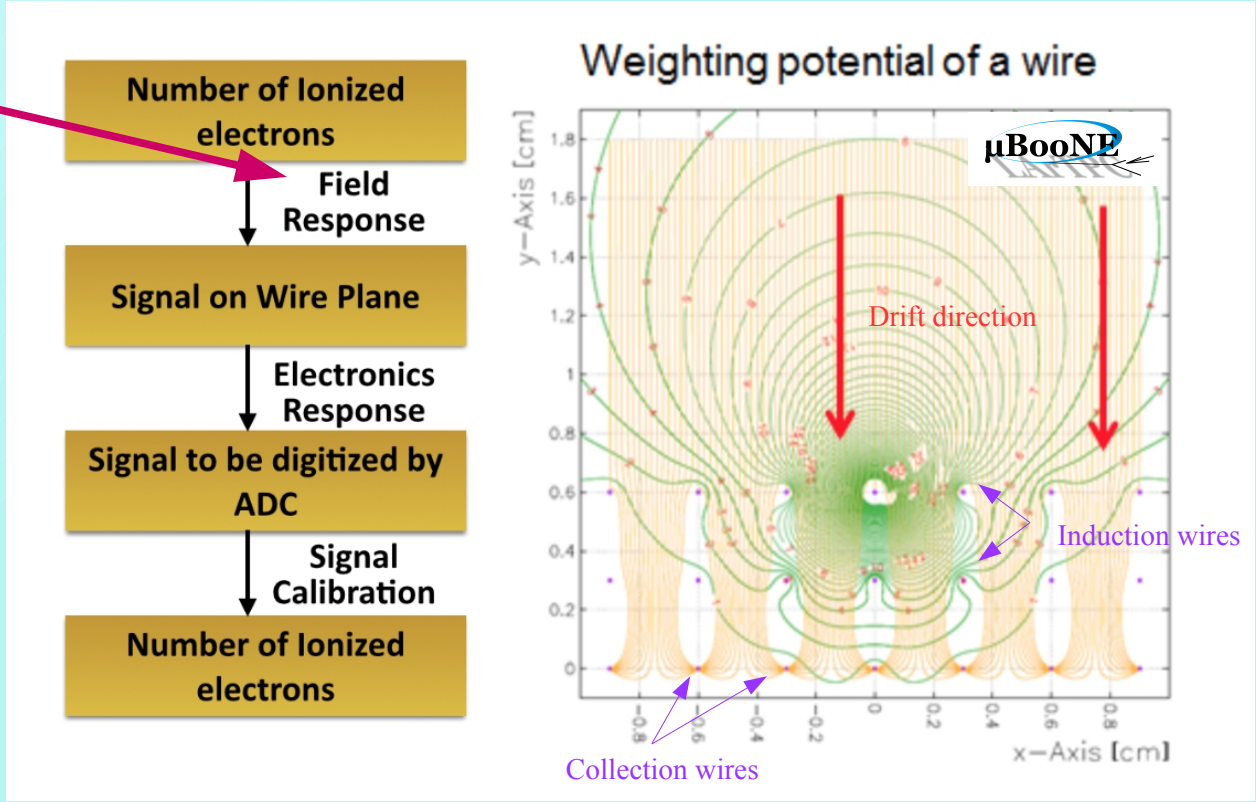
From MicroBooNE-Note-1017-pub

TPC signal formation

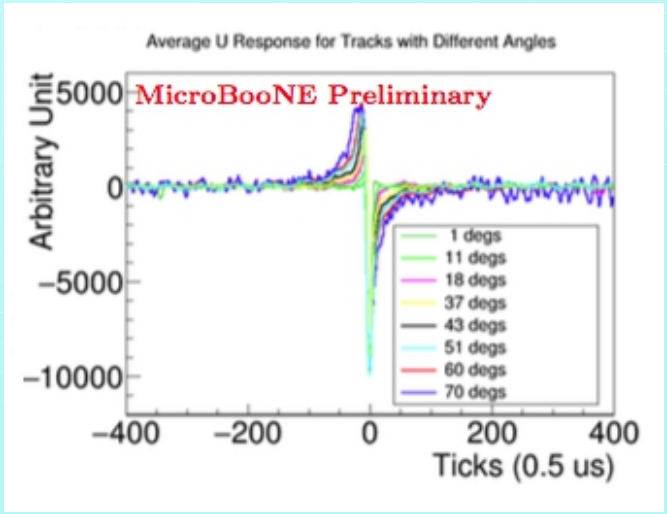
Induced current $i \propto q\, \vec{E}_w \cdot \vec{v}_q$

$\vec{E}_w$ = "weighting E field"

$\vec{v}_q$ = velocity of charge 'q'



The signal shape also depends on the track angle

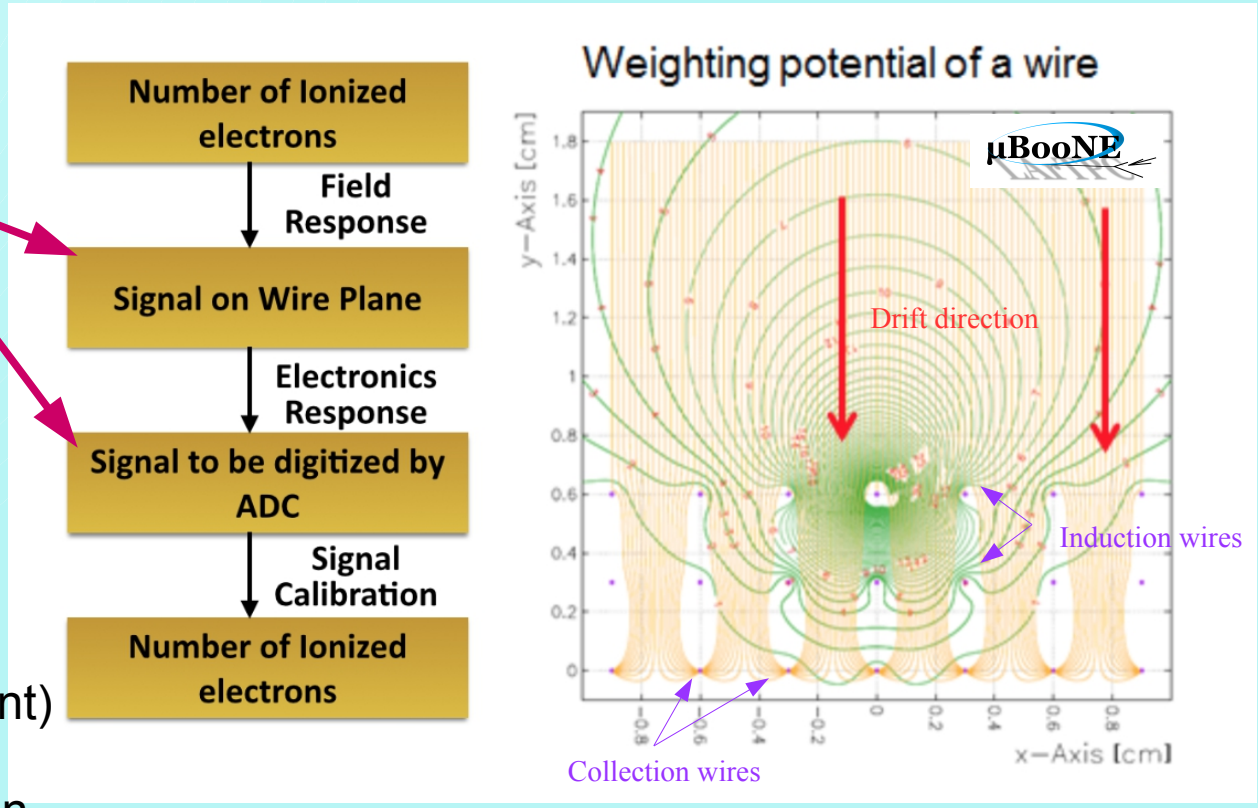


From MicroBooNE-Note-1017-pub

Will come back to this point...

TPC signal formation

Noise injection



From MicroBooNE-Note-1017-pub

MicroBooNE example

Identified sources of noise

- First input transistor
- Warm shaping amplifier (small)
- Other readout circuits (coherent)
- Wire bias power supplies (coherent)
- Cathode HV (coherent)
- ASIC saturation due to wire motion

Treat coherent noise with specialized algorithms
Optimal filter applicable (in principle) to remaining sources during **deconvolution**

Deconvolution

- Given

$$y(t) = (h * x)(t) + n(t)$$

  - where $y(t)$ = **measured output signal** (raw digits)

    $(h * x)(t)$ = convolution of impulse response $h(t)$ and (unknown) input signal $x(t)$

    $n(t)$ = (unknown) noise

- Optimal signal estimate (minimum mean squared error) given only
  - Finite **impulse response** of the front-end electronics
  - Estimated **mean power spectrum** for the **signal** and the **noise**

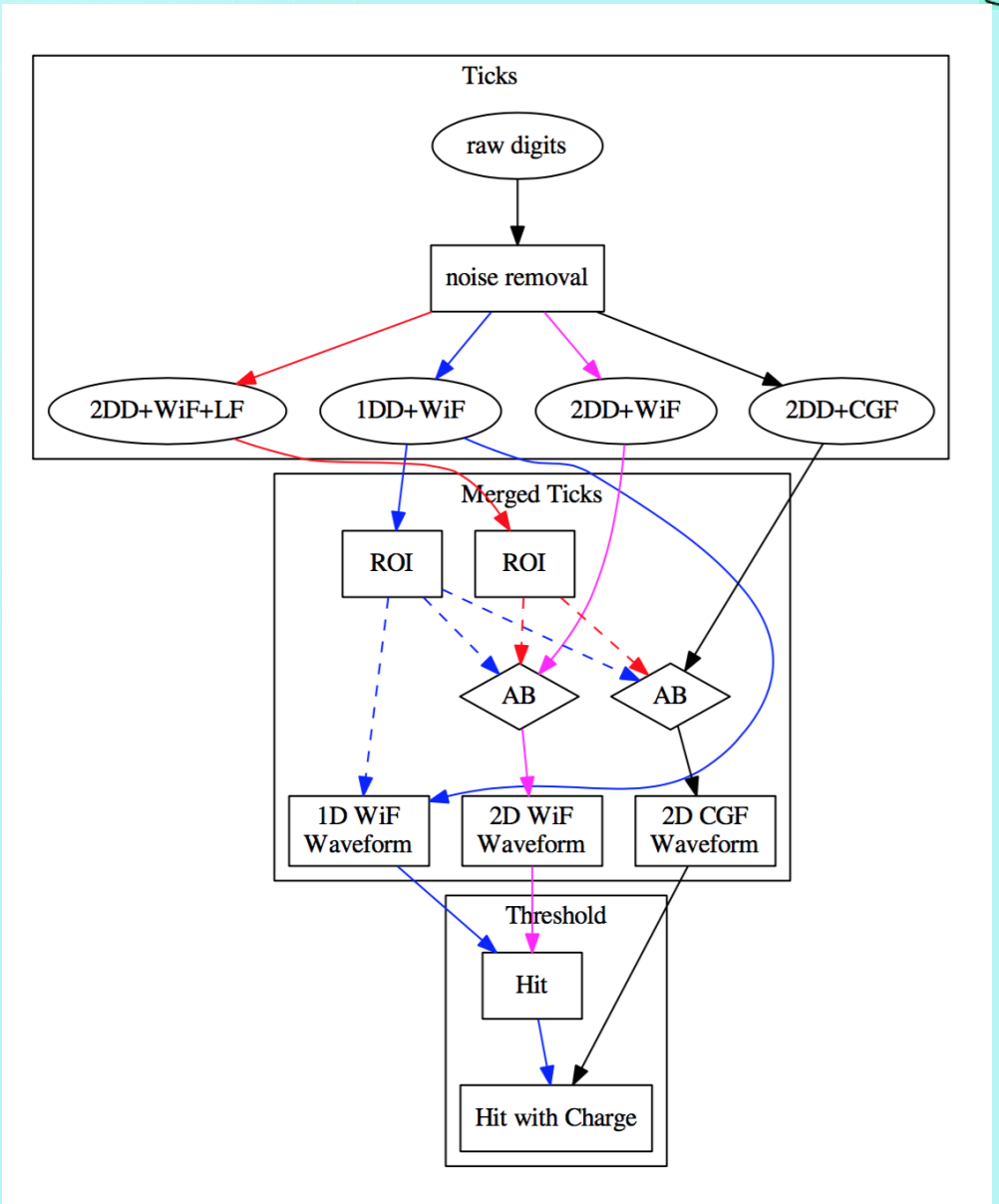
  (used in a Weiner filter)

  assuming stationary signal and noise power spectra

- But signal spectrum
  - depends on track angle
  - signals on nearby tracks

In practice, the deconvolution procedure is somewhat complex

An example from MicroBooNE

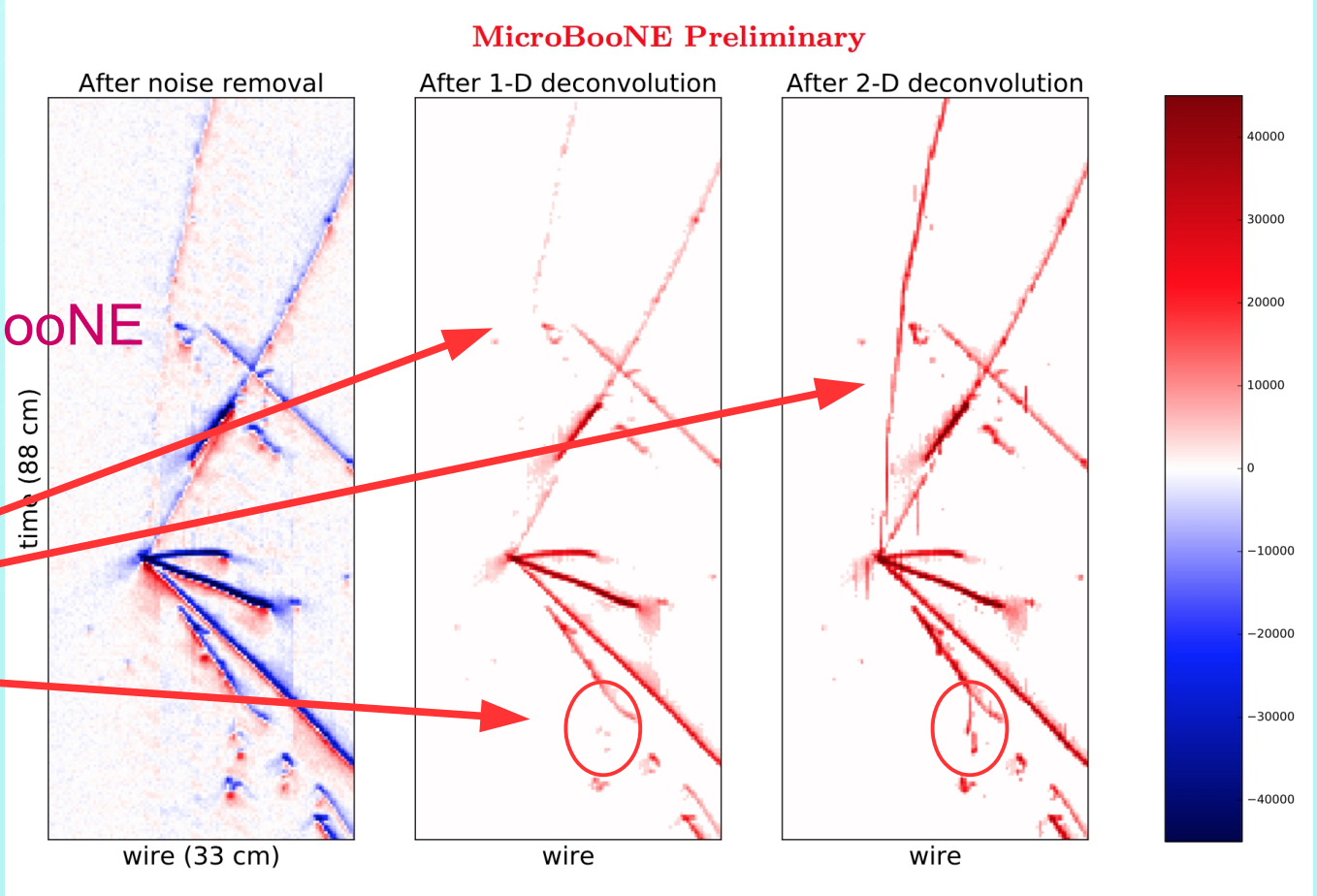


From MicroBooNE-Note-1017-pub

In practice, the deconvolution procedure is somewhat complex

An example from MicroBooNE

It pays to get this right!!



From MicroBooNE-Note-1017-pub

Wire calibration and deconvolution

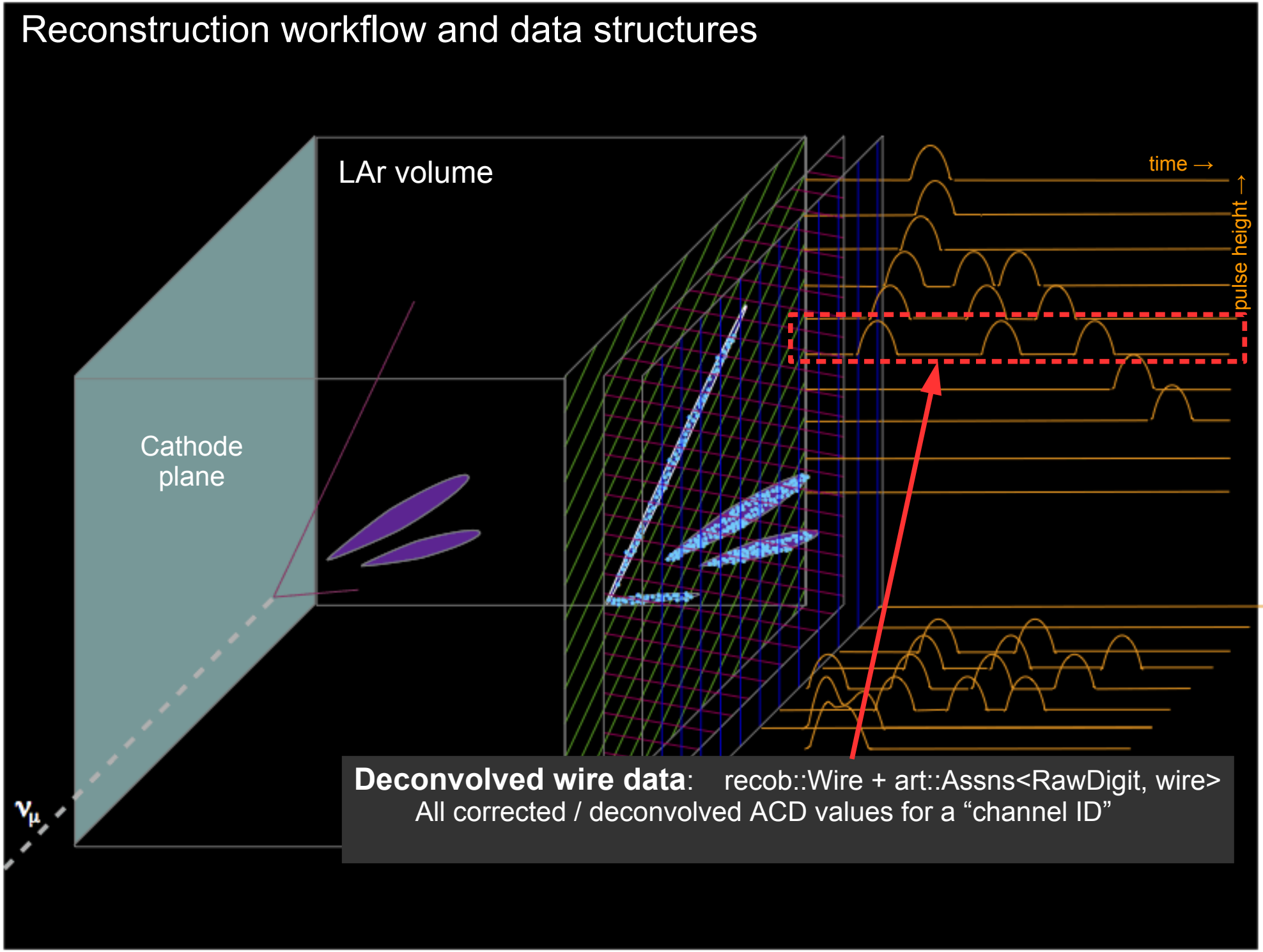
- Calibration
  - Only pedestal subtraction at this phase.
    (Channel gains come later...)
- Deconvolution

Performed by a number of classes

CalWire ...
CalROI ...
SignalShapingService ... (1D)
SignalShaping
LArFFT
Wire-cell (2D)

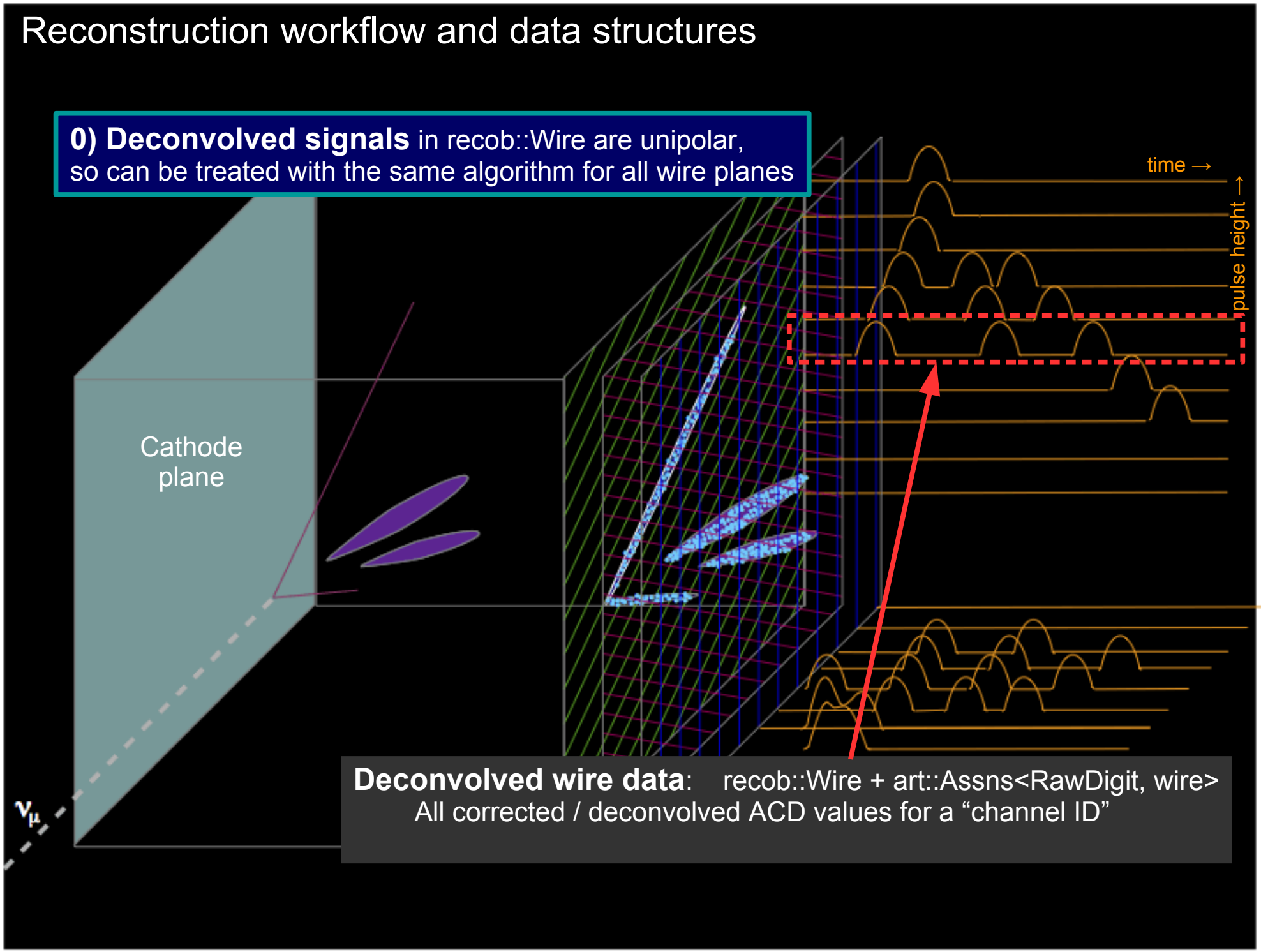
(Experiment-specific)

Reconstruction workflow and data structures



Reconstruction workflow and data structures

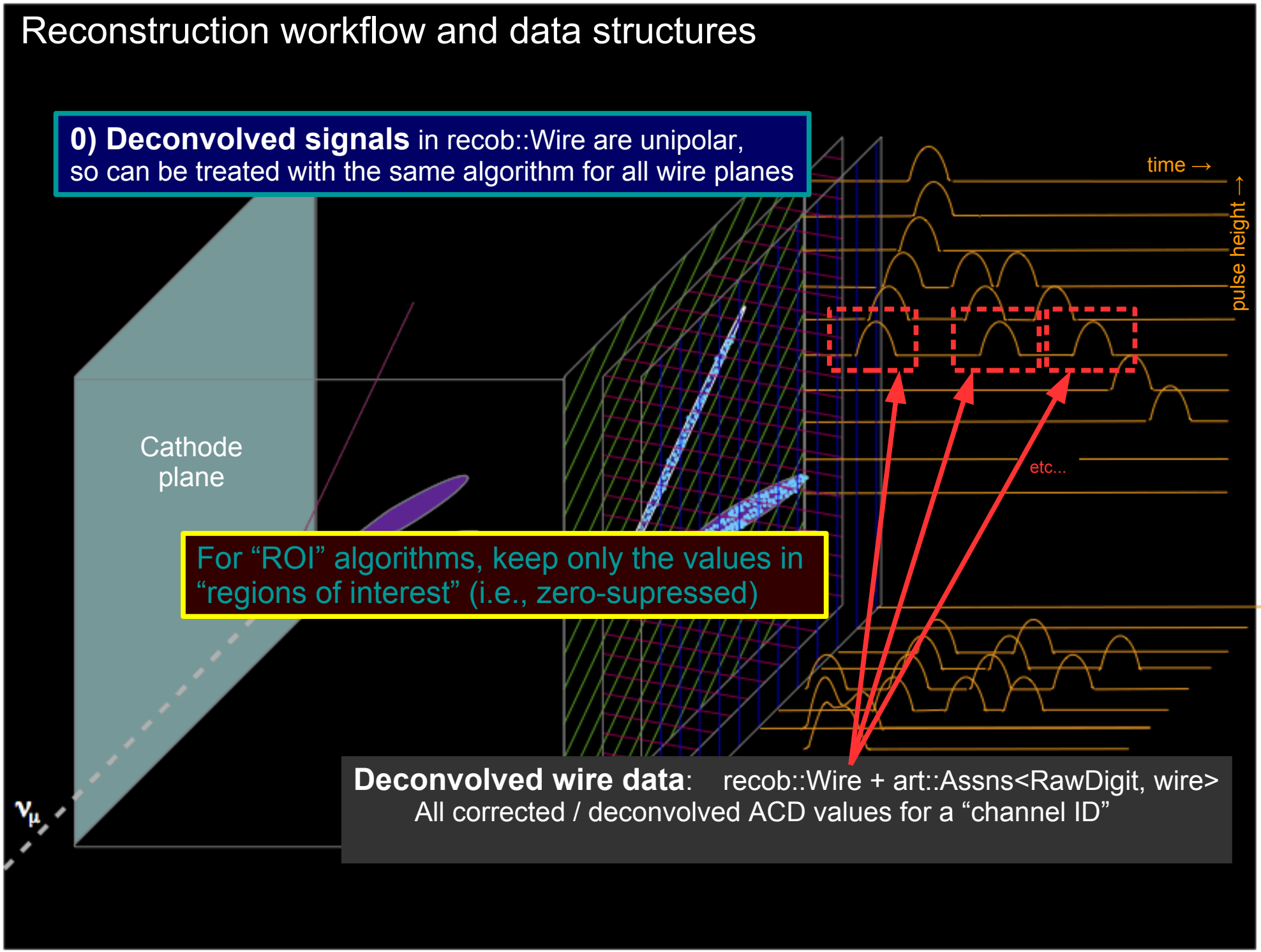
0) Deconvolved signals in recob::Wire are unipolar, so can be treated with the same algorithm for all wire planes



Deconvolved wire data: recob::Wire + art::Assns<RawDigit, wire>
All corrected / deconvolved ACD values for a “channel ID”

Reconstruction workflow and data structures

0) Deconvolved signals in recob::Wire are unipolar, so can be treated with the same algorithm for all wire planes



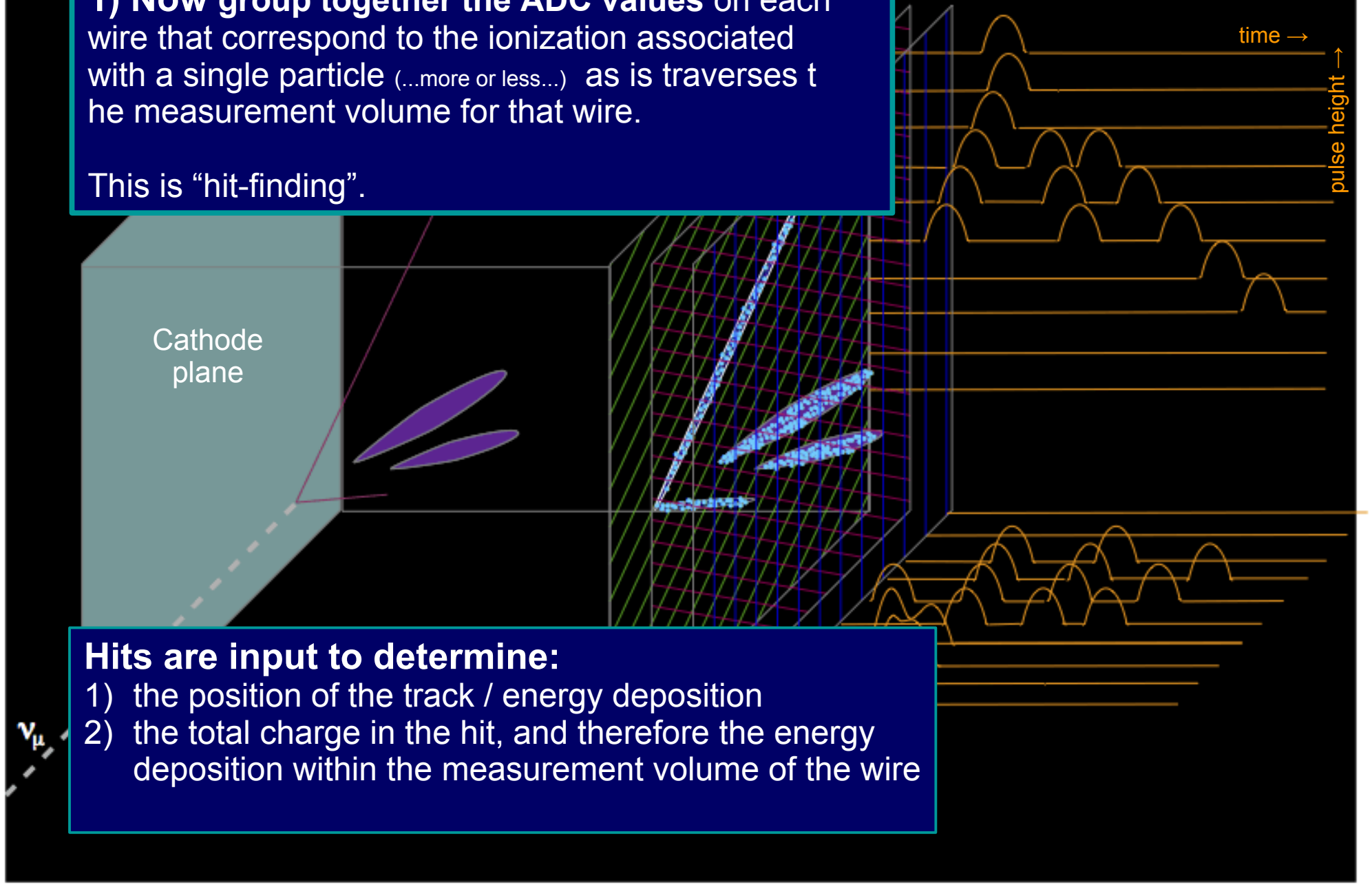
For "ROI" algorithms, keep only the values in "regions of interest" (i.e., zero-supressed)

Deconvolved wire data: recob::Wire + art::Assns<RawDigit, wire>
All corrected / deconvolved ACD values for a "channel ID"

Reconstruction workflow and data structures

1) Now group together the ADC values on each wire that correspond to the ionization associated with a single particle (...more or less...) as is traverses t he measurement volume for that wire.

This is “hit-finding”.



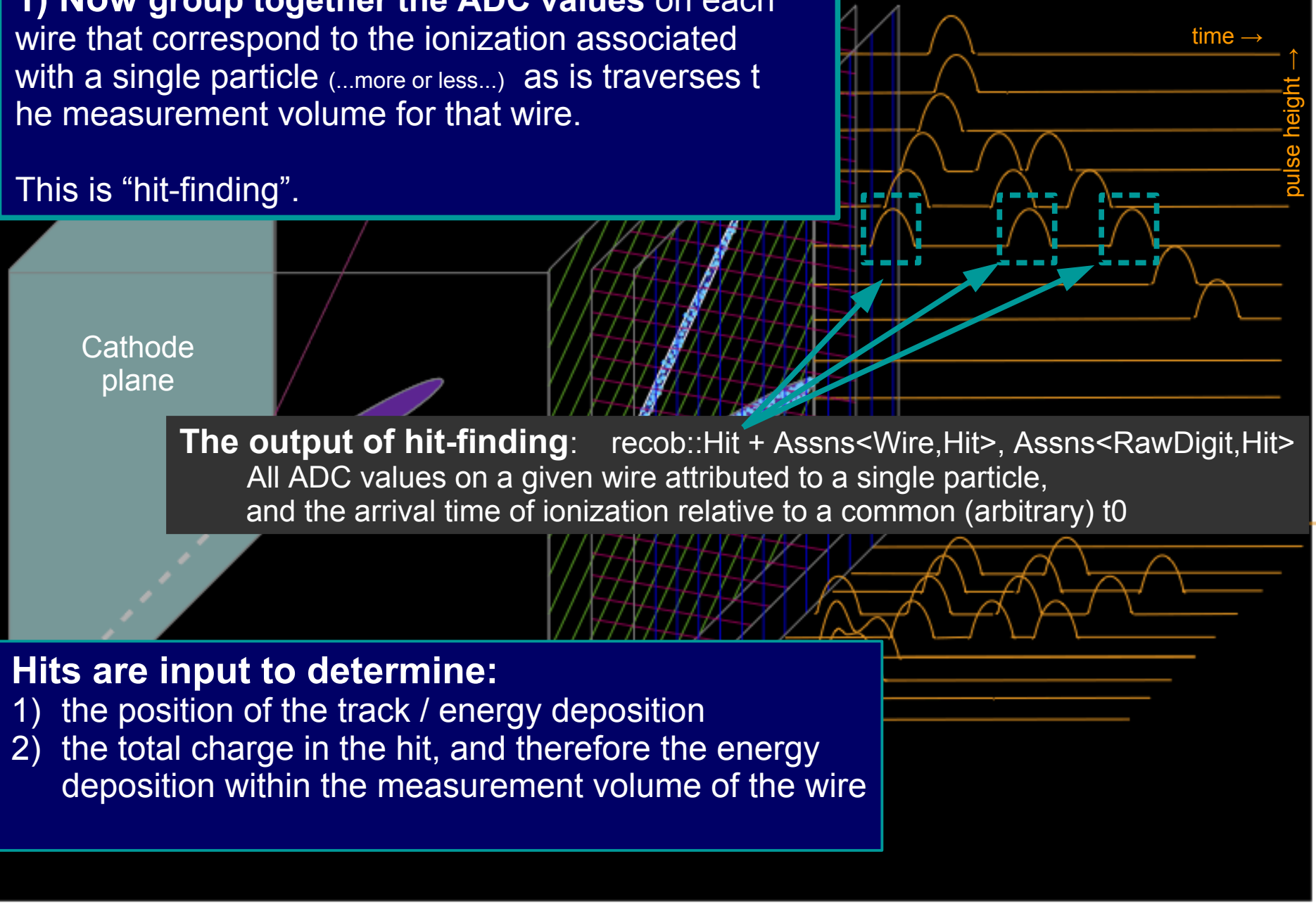
Hits are input to determine:

1) the position of the track / energy deposition
2) the total charge in the hit, and therefore the energy deposition within the measurement volume of the wire

Reconstruction workflow and data structures

1) Now group together the ADC values on each wire that correspond to the ionization associated with a single particle (...more or less...) as is traverses t he measurement volume for that wire.

This is “hit-finding”.



The output of hit-finding: recob::Hit + Assns<Wire,Hit>, Assns<RawDigit,Hit>
All ADC values on a given wire attributed to a single particle,
and the arrival time of ionization relative to a common (arbitrary) t0

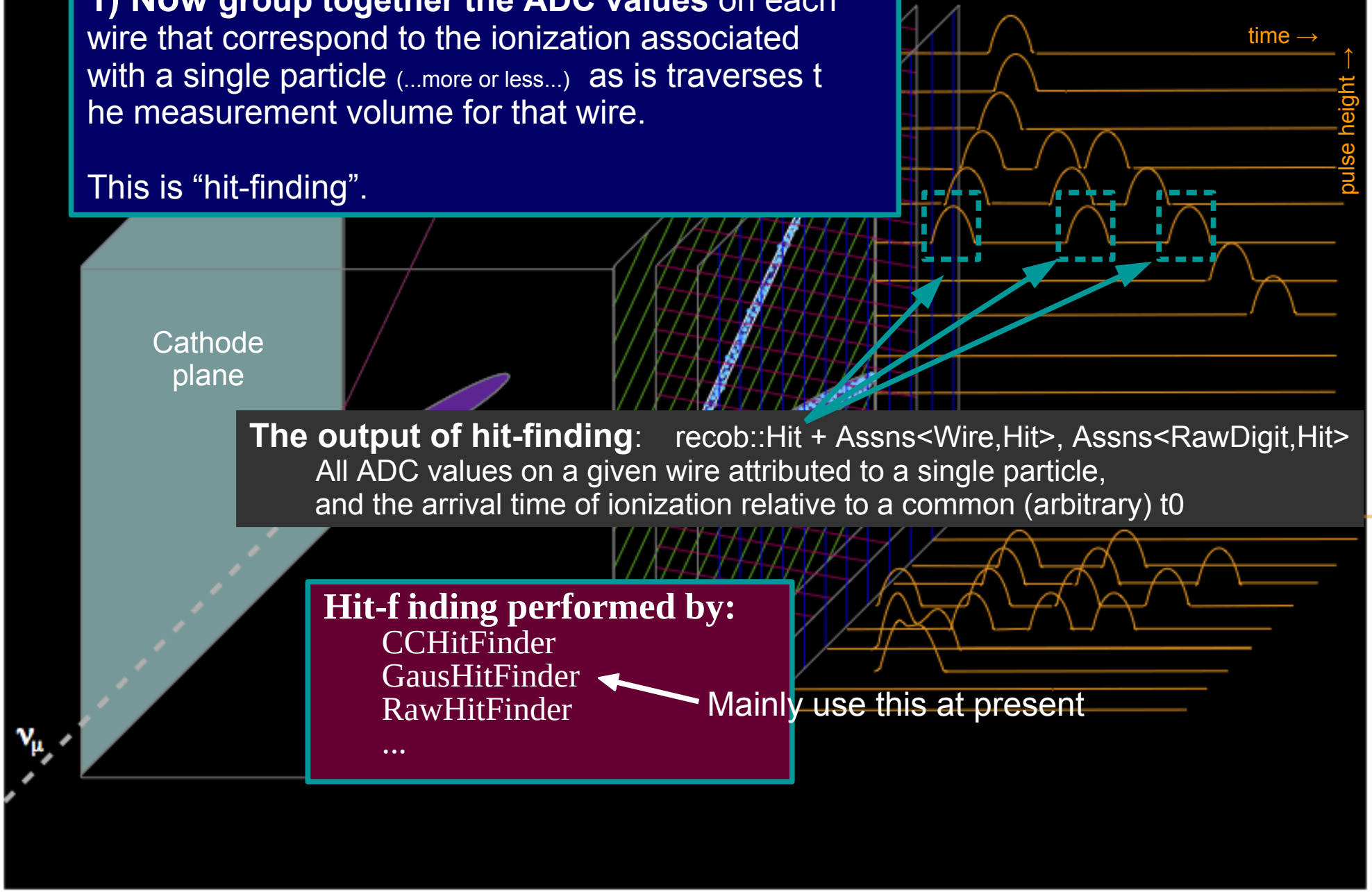
Hits are input to determine:

1) the position of the track / energy deposition
2) the total charge in the hit, and therefore the energy deposition within the measurement volume of the wire

Reconstruction workflow and data structures

1) Now group together the ADC values on each wire that correspond to the ionization associated with a single particle (...more or less...) as is traverses t he measurement volume for that wire.

This is “hit-finding”.



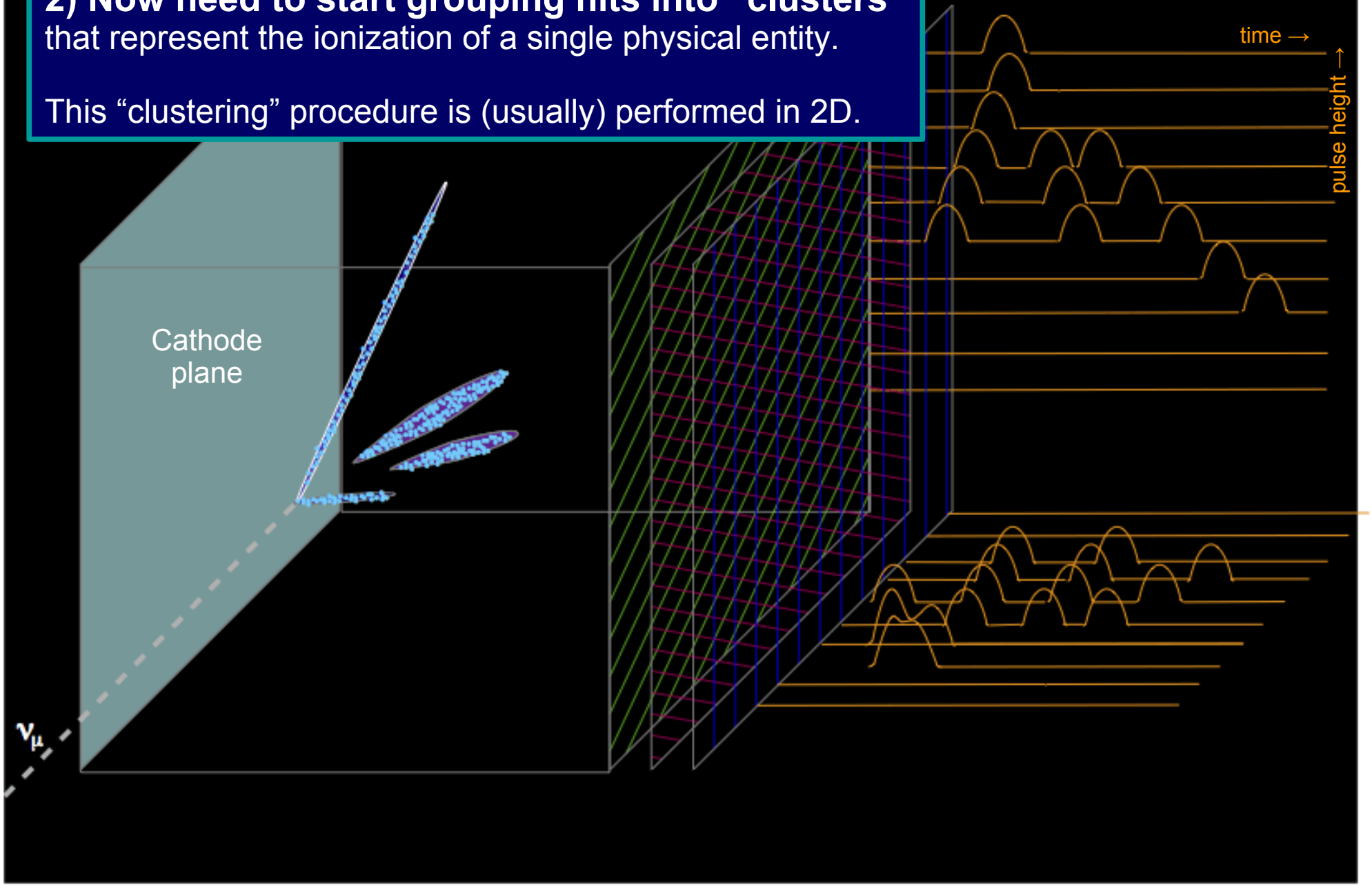
The output of hit-finding: recob::Hit + Assns<Wire,Hit>, Assns<RawDigit,Hit>
All ADC values on a given wire attributed to a single particle,
and the arrival time of ionization relative to a common (arbitrary) t0

Hit-f inding performed by:
- CCHitFinder
- GausHitFinder ← Mainly use this at present
- RawHitFinder
- ...

Reconstruction workflow and data structures

2) Now need to start grouping hits into "clusters" that represent the ionization of a single physical entity.

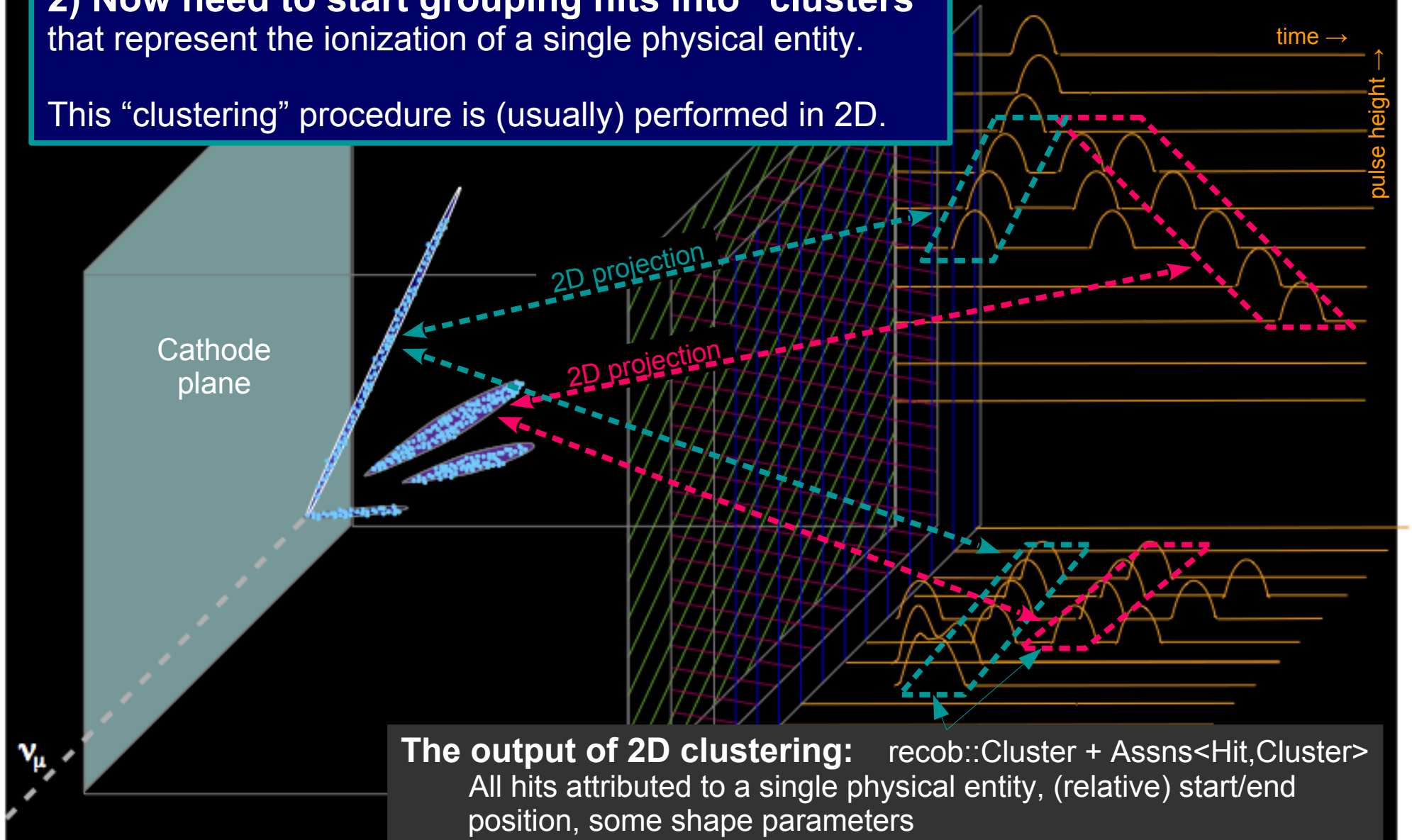
This "clustering" procedure is (usually) performed in 2D.



Reconstruction workflow and data structures

2) Now need to start grouping hits into "clusters" that represent the ionization of a single physical entity.

This "clustering" procedure is (usually) performed in 2D.

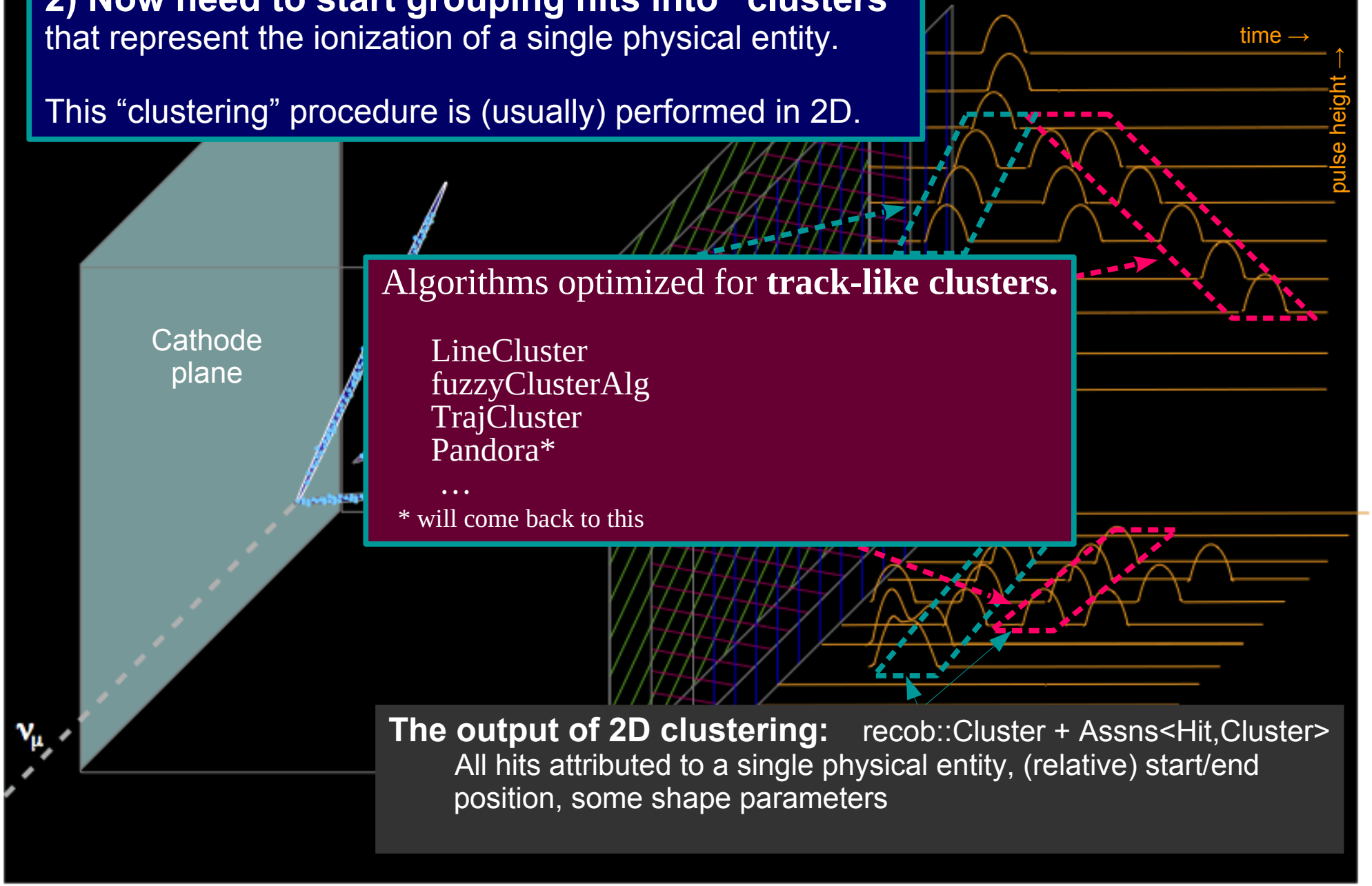


The output of 2D clustering: recob::Cluster + Assns<Hit,Cluster>
All hits attributed to a single physical entity, (relative) start/end position, some shape parameters

Reconstruction workflow and data structures

2) Now need to start grouping hits into "clusters" that represent the ionization of a single physical entity.

This "clustering" procedure is (usually) performed in 2D.



Algorithms optimized for **track-like clusters.**

LineCluster
fuzzyClusterAlg
TrajCluster
Pandora*
…

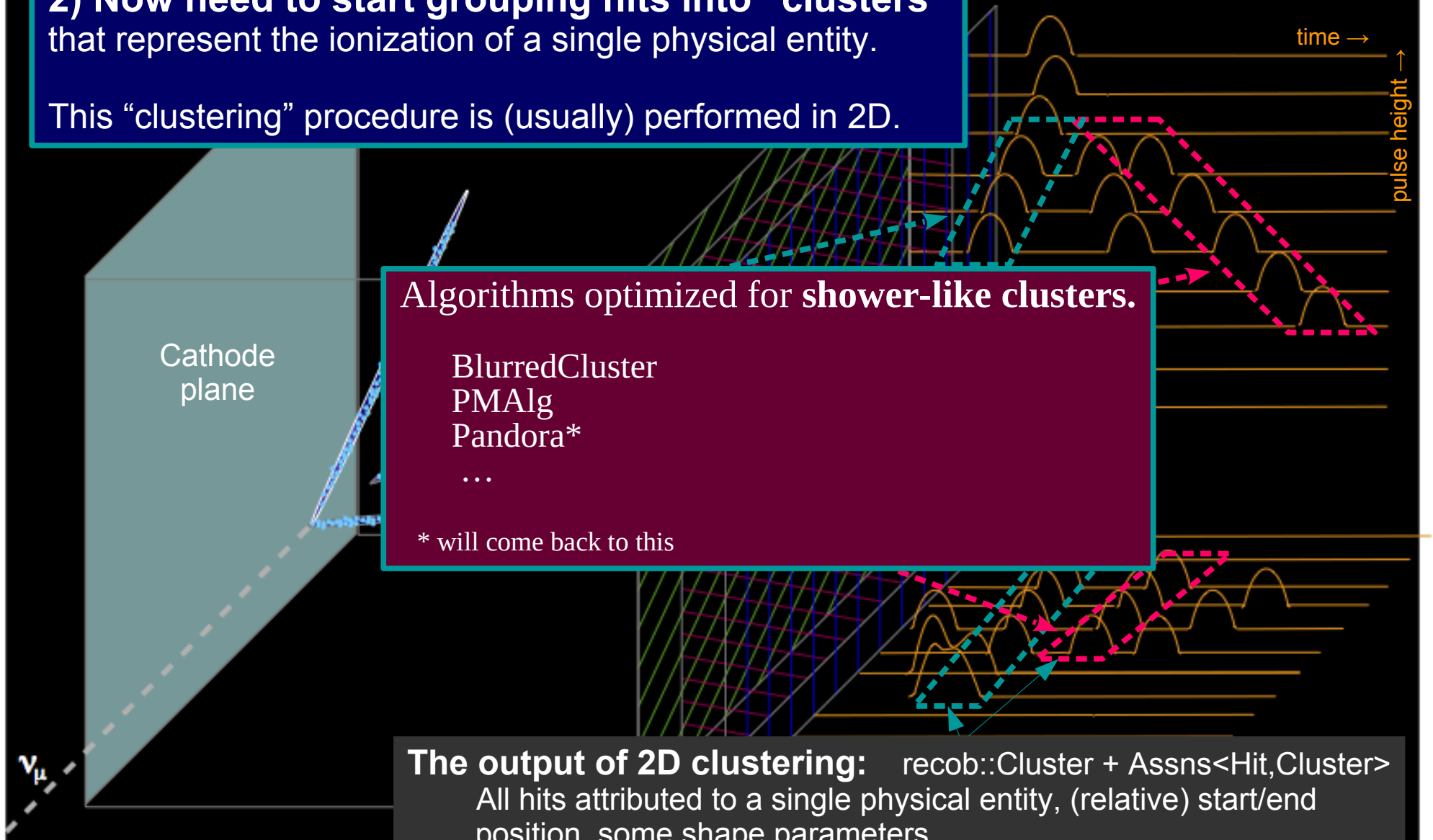
* will come back to this

The output of 2D clustering: recob::Cluster + Assns<Hit,Cluster>
All hits attributed to a single physical entity, (relative) start/end position, some shape parameters

Reconstruction workflow and data structures

2) Now need to start grouping hits into "clusters" that represent the ionization of a single physical entity.

This "clustering" procedure is (usually) performed in 2D.



Algorithms optimized for **shower-like clusters.**

- BlurredCluster
- PMAlg
- Pandora*
- …

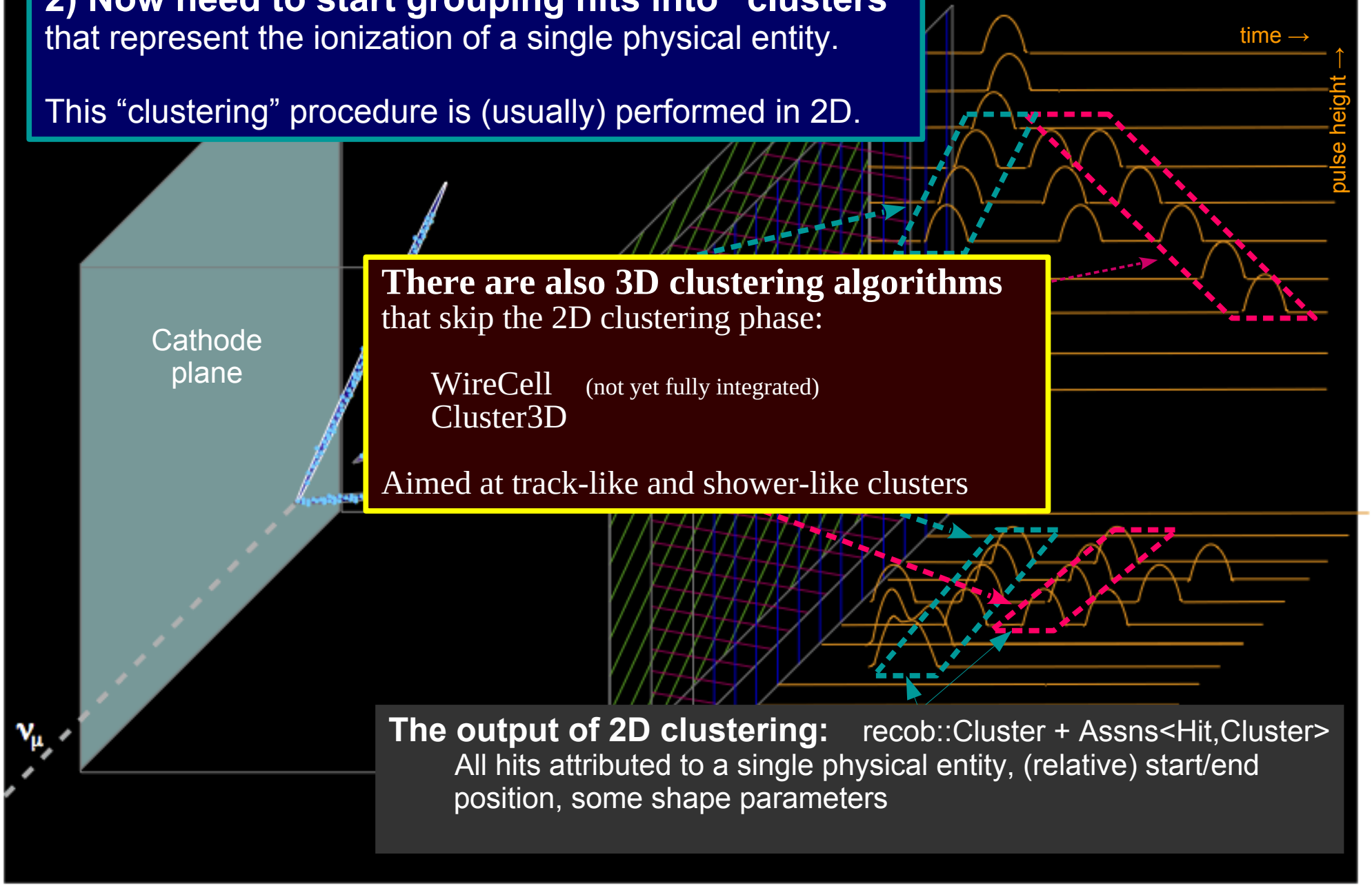
* will come back to this

The output of 2D clustering: recob::Cluster + Assns<Hit,Cluster>
All hits attributed to a single physical entity, (relative) start/end position, some shape parameters

Reconstruction workflow and data structures

2) Now need to start grouping hits into "clusters" that represent the ionization of a single physical entity.

This "clustering" procedure is (usually) performed in 2D.



There are also 3D clustering algorithms that skip the 2D clustering phase:

WireCell (not yet fully integrated)
Cluster3D

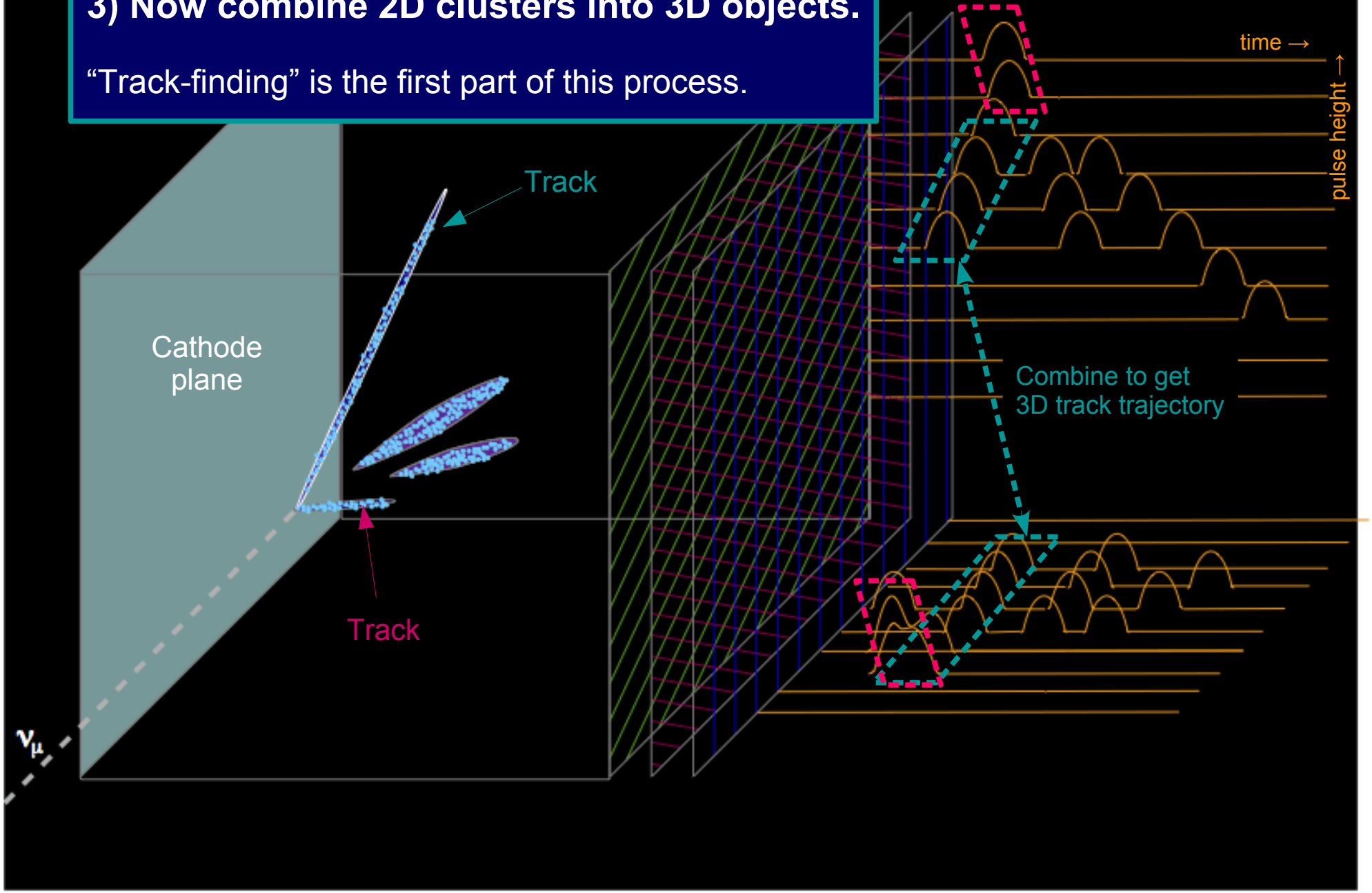
Aimed at track-like and shower-like clusters

The output of 2D clustering: recob::Cluster + Assns<Hit,Cluster>
All hits attributed to a single physical entity, (relative) start/end position, some shape parameters

Reconstruction workflow and data structures

3) Now combine 2D clusters into 3D objects.

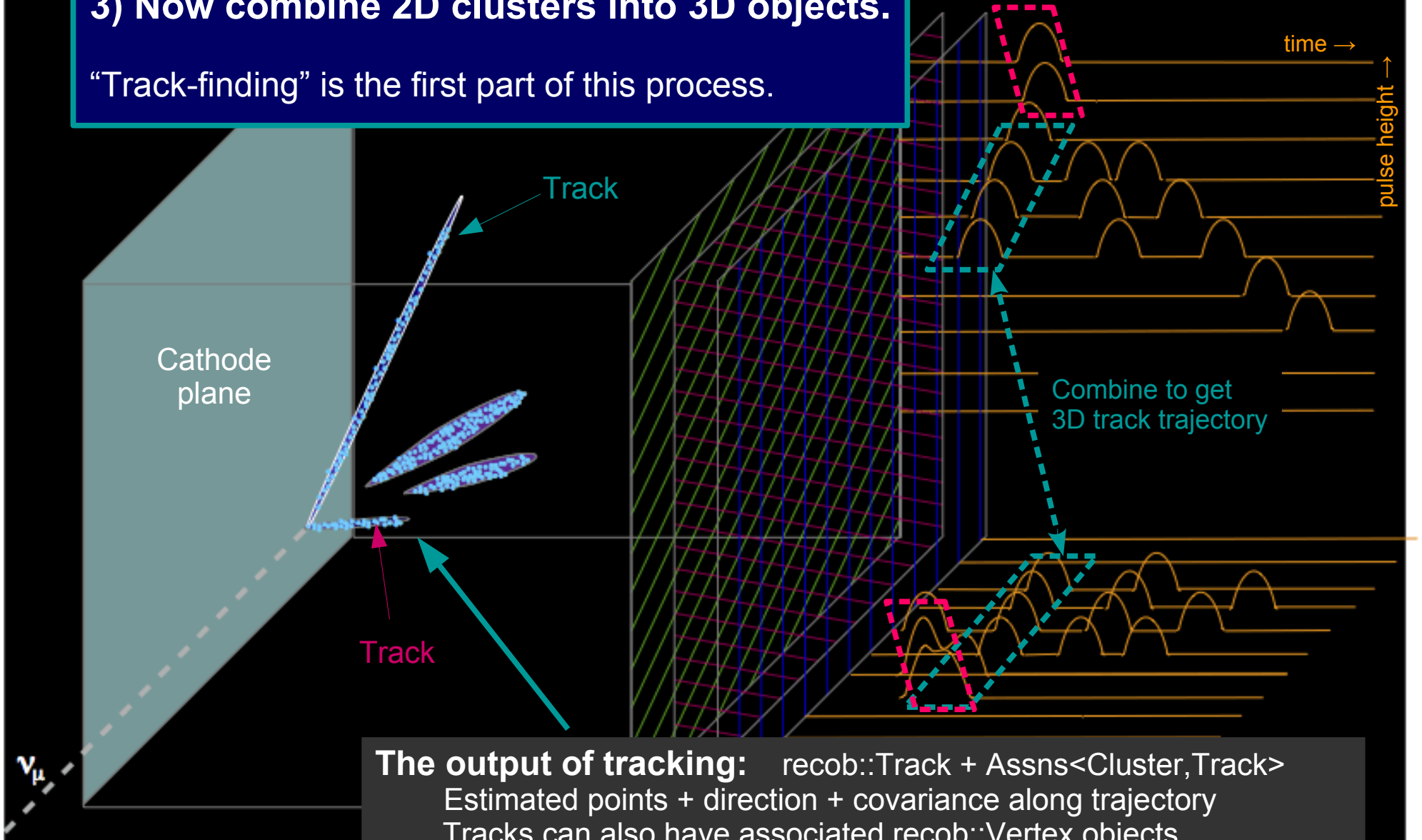
"Track-finding" is the first part of this process.



Reconstruction workflow and data structures

3) Now combine 2D clusters into 3D objects.

"Track-finding" is the first part of this process.

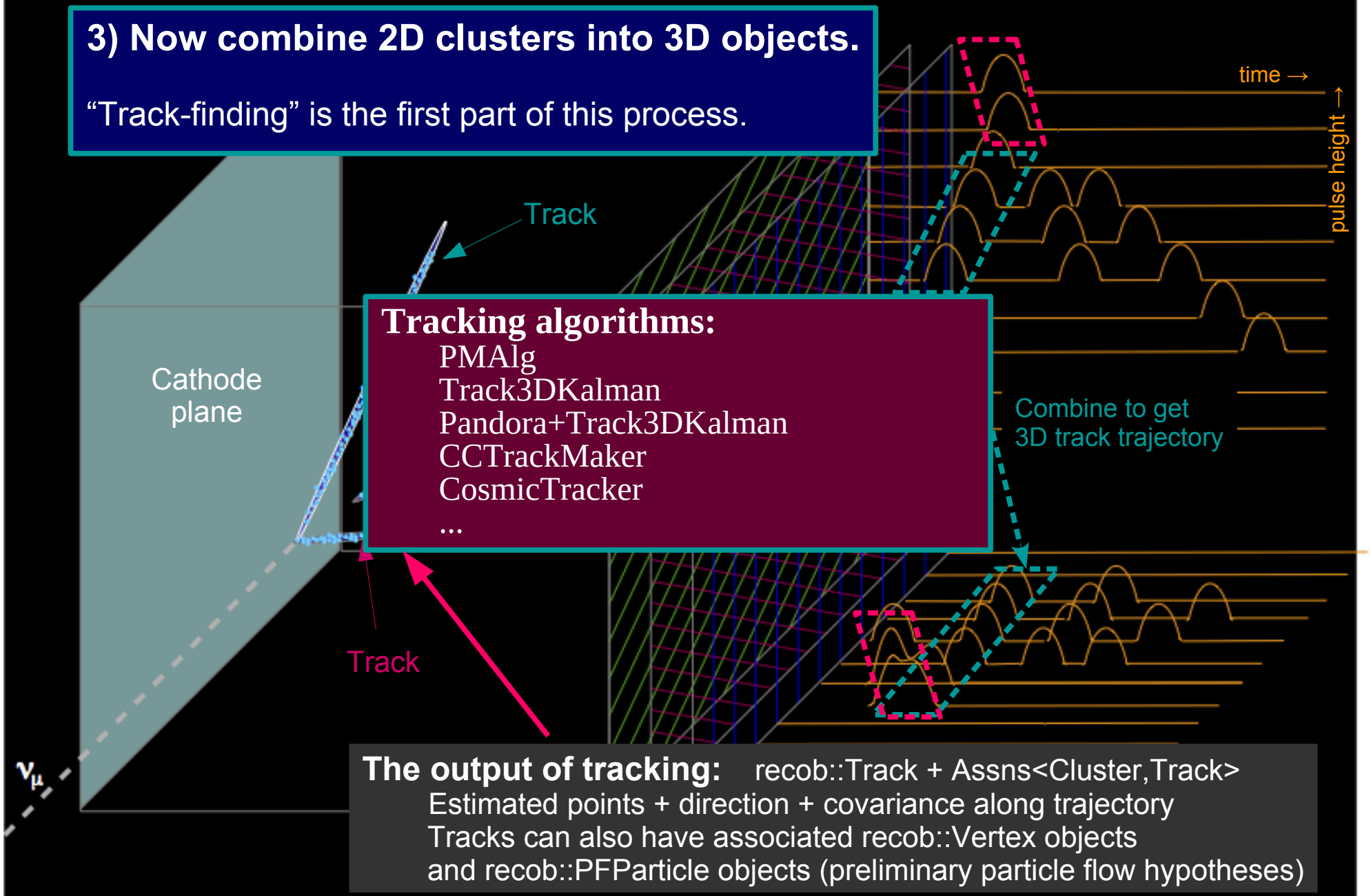


The output of tracking: recob::Track + Assns<Cluster,Track>
Estimated points + direction + covariance along trajectory
Tracks can also have associated recob::Vertex objects
and recob::PFParticle objects (preliminary particle flow hypotheses)

Reconstruction workflow and data structures

3) Now combine 2D clusters into 3D objects.

"Track-finding" is the first part of this process.



Tracking algorithms:

- PMAlg
- Track3DKalman
- Pandora+Track3DKalman
- CCTrackMaker
- CosmicTracker
- ...

The output of tracking: recob::Track + Assns<Cluster,Track>

Estimated points + direction + covariance along trajectory

Tracks can also have associated recob::Vertex objects

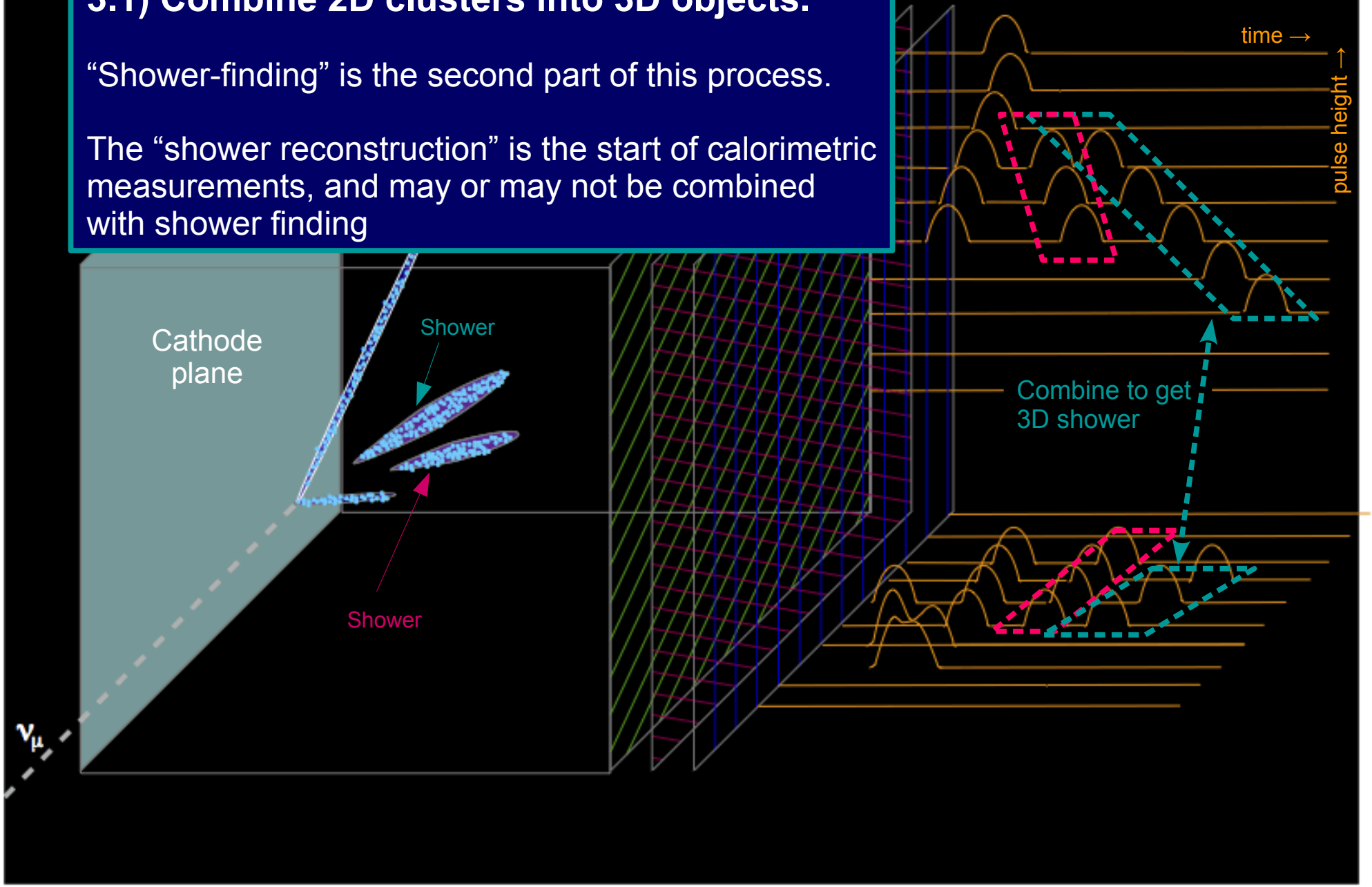
and recob::PFParticle objects (preliminary particle flow hypotheses)

Reconstruction workflow and data structures

3.1) Combine 2D clusters into 3D objects.

"Shower-finding" is the second part of this process.

The "shower reconstruction" is the start of calorimetric measurements, and may or may not be combined with shower finding

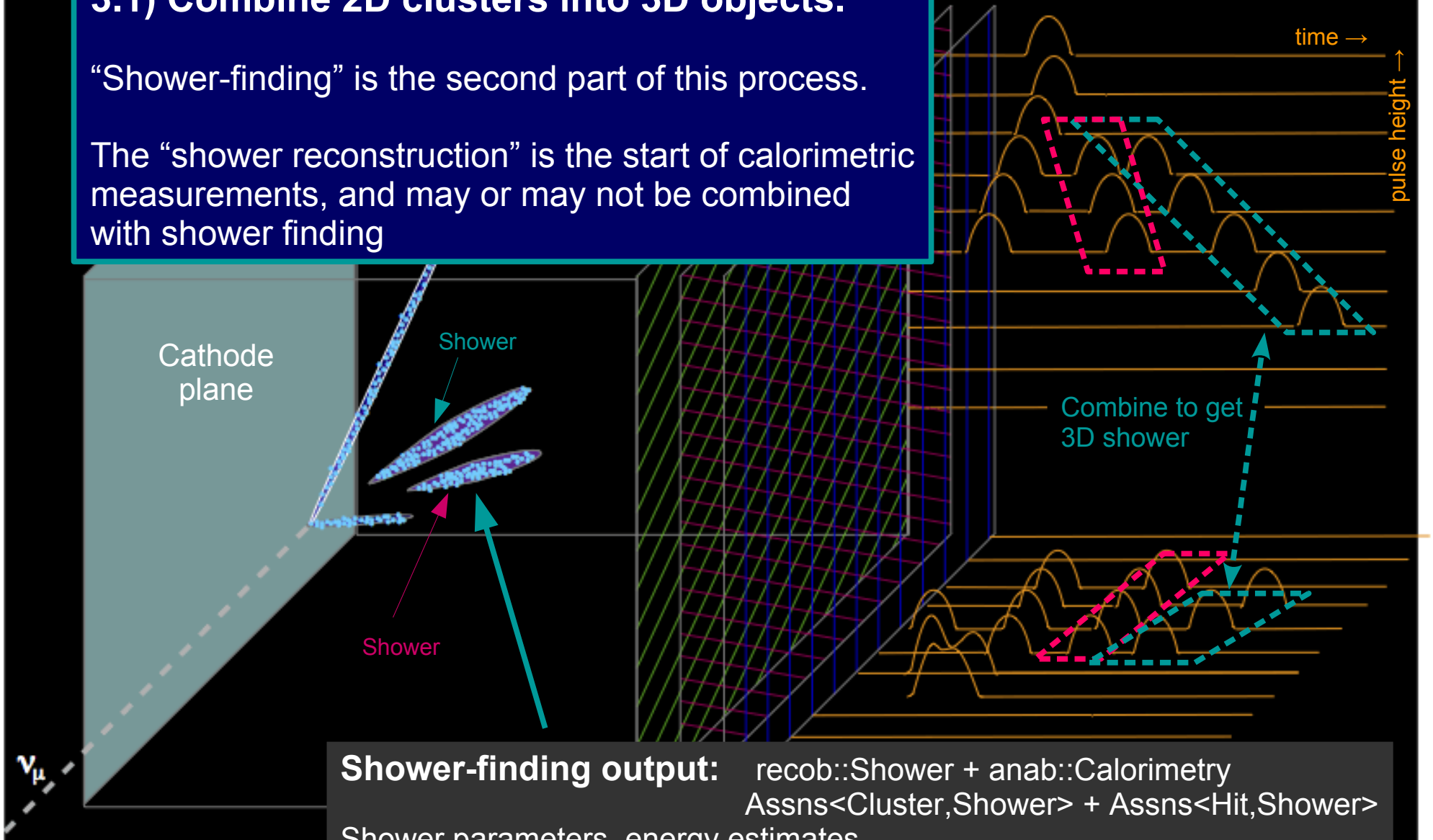


Reconstruction workflow and data structures

3.1) Combine 2D clusters into 3D objects.

"Shower-finding" is the second part of this process.

The "shower reconstruction" is the start of calorimetric measurements, and may or may not be combined with shower finding



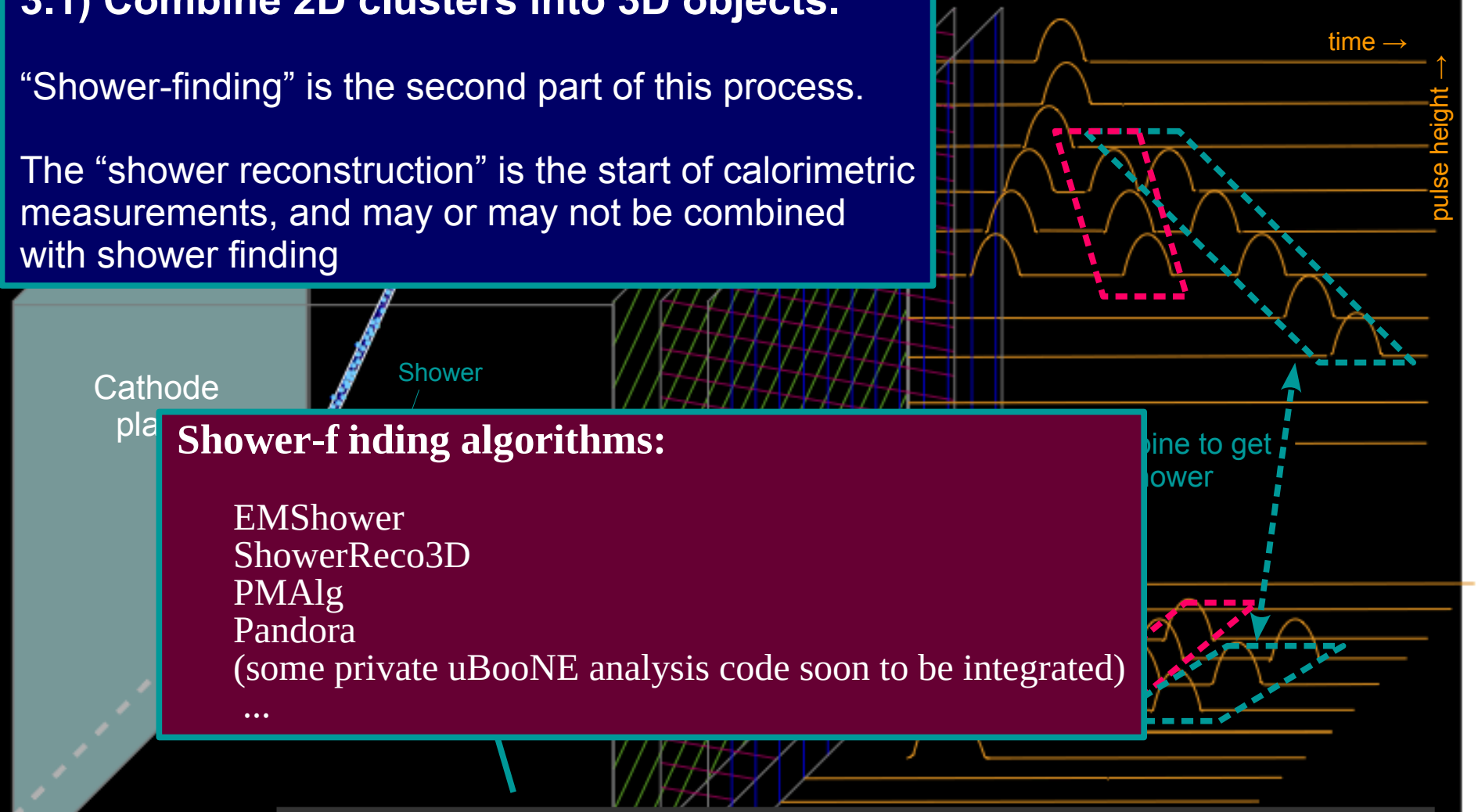
Shower-finding output: recob::Shower + anab::Calorimetry
Assns<Cluster,Shower> + Assns<Hit,Shower>
Shower parameters, energy estimates

Reconstruction workflow and data structures

3.1) Combine 2D clusters into 3D objects.

"Shower-finding" is the second part of this process.

The "shower reconstruction" is the start of calorimetric measurements, and may or may not be combined with shower finding



Shower-f nding algorithms:

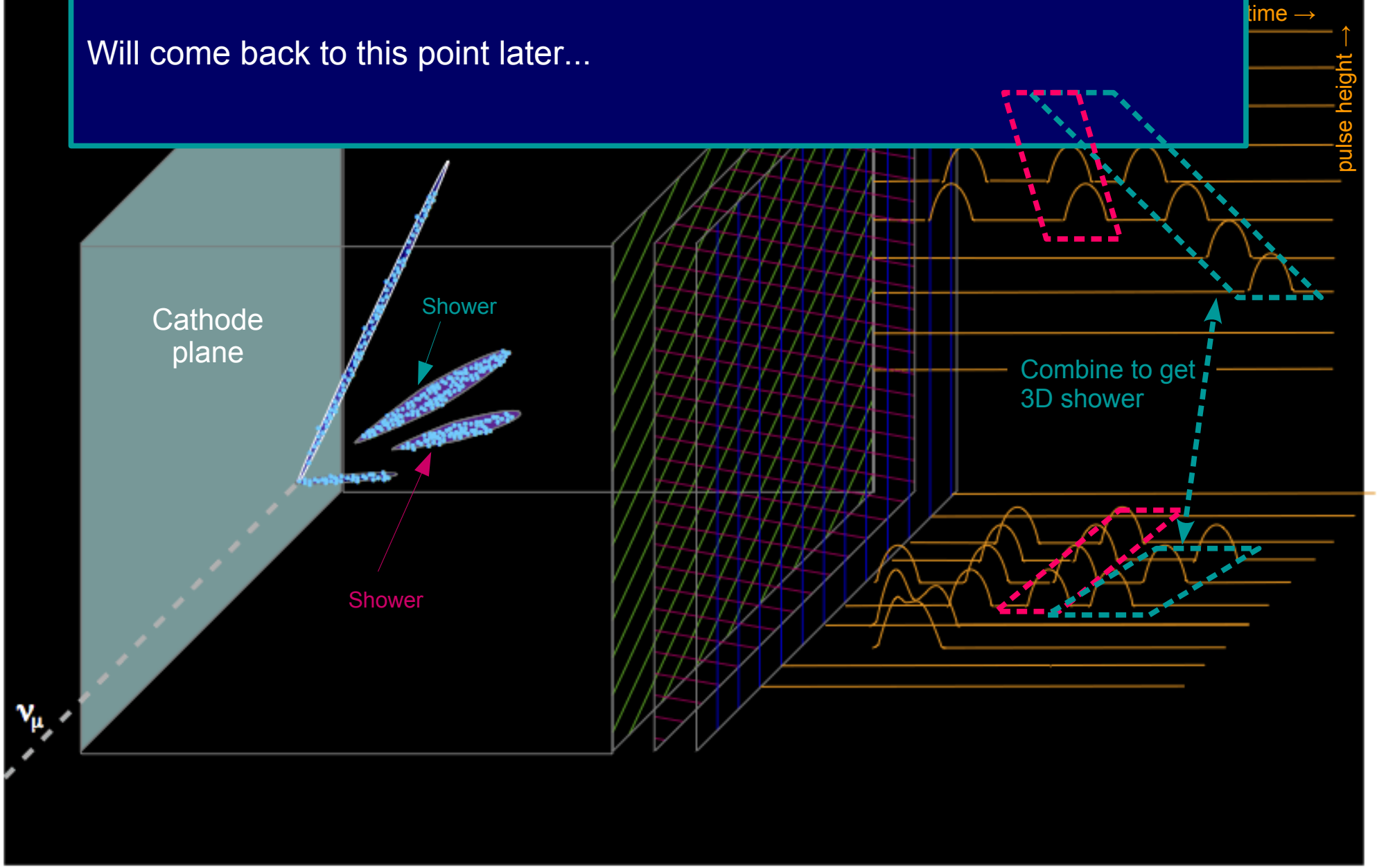
EMShower
ShowerReco3D
PMAlg
Pandora
(some private uBooNE analysis code soon to be integrated)
...

Shower-finding output: recob::Shower + anab::Calorimetry
Assns<Cluster,Shower> + Assns<Hit,Shower>
Shower parameters, energy estimates

Reconstruction workflow and data structures

3.2) Can assemble 2D clusters into hierarchies of 3D objects

Will come back to this point later...

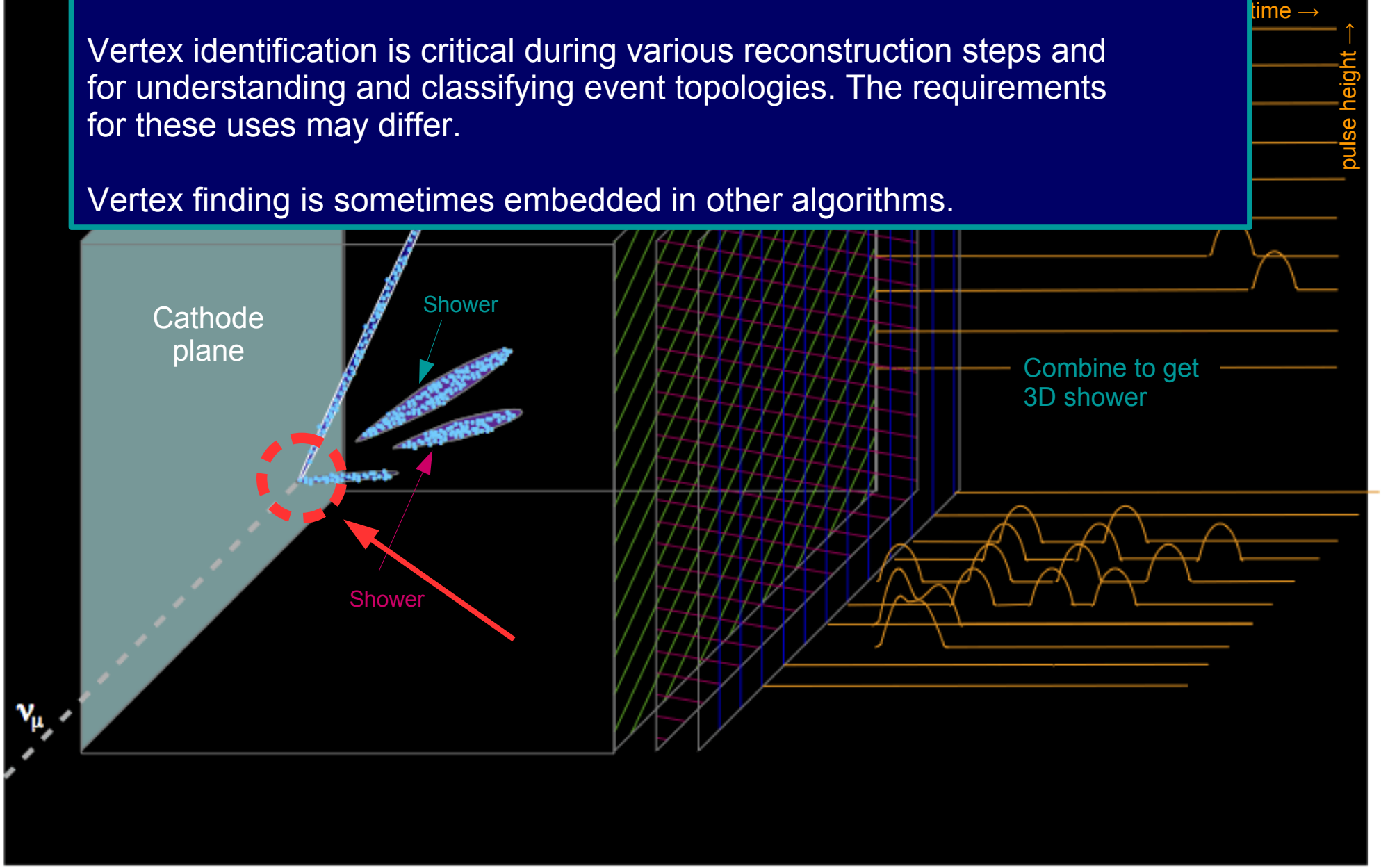


Reconstruction workflow and data structures

4) Identify vertices in the event from tracks in the event

Vertex identification is critical during various reconstruction steps and for understanding and classifying event topologies. The requirements for these uses may differ.

Vertex finding is sometimes embedded in other algorithms.

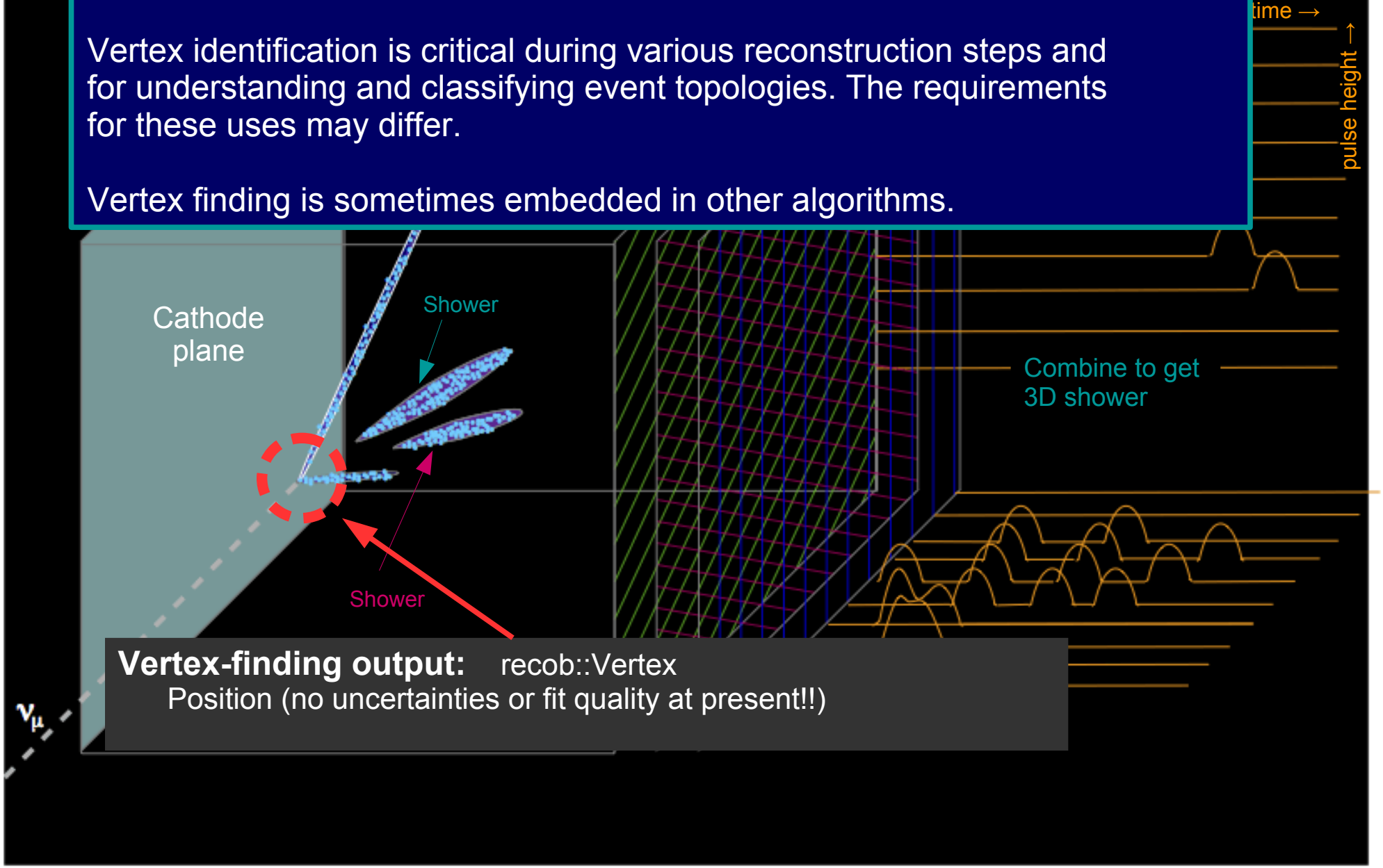


Reconstruction workflow and data structures

4) Identify vertices in the event from tracks in the event

Vertex identification is critical during various reconstruction steps and for understanding and classifying event topologies. The requirements for these uses may differ.

Vertex finding is sometimes embedded in other algorithms.



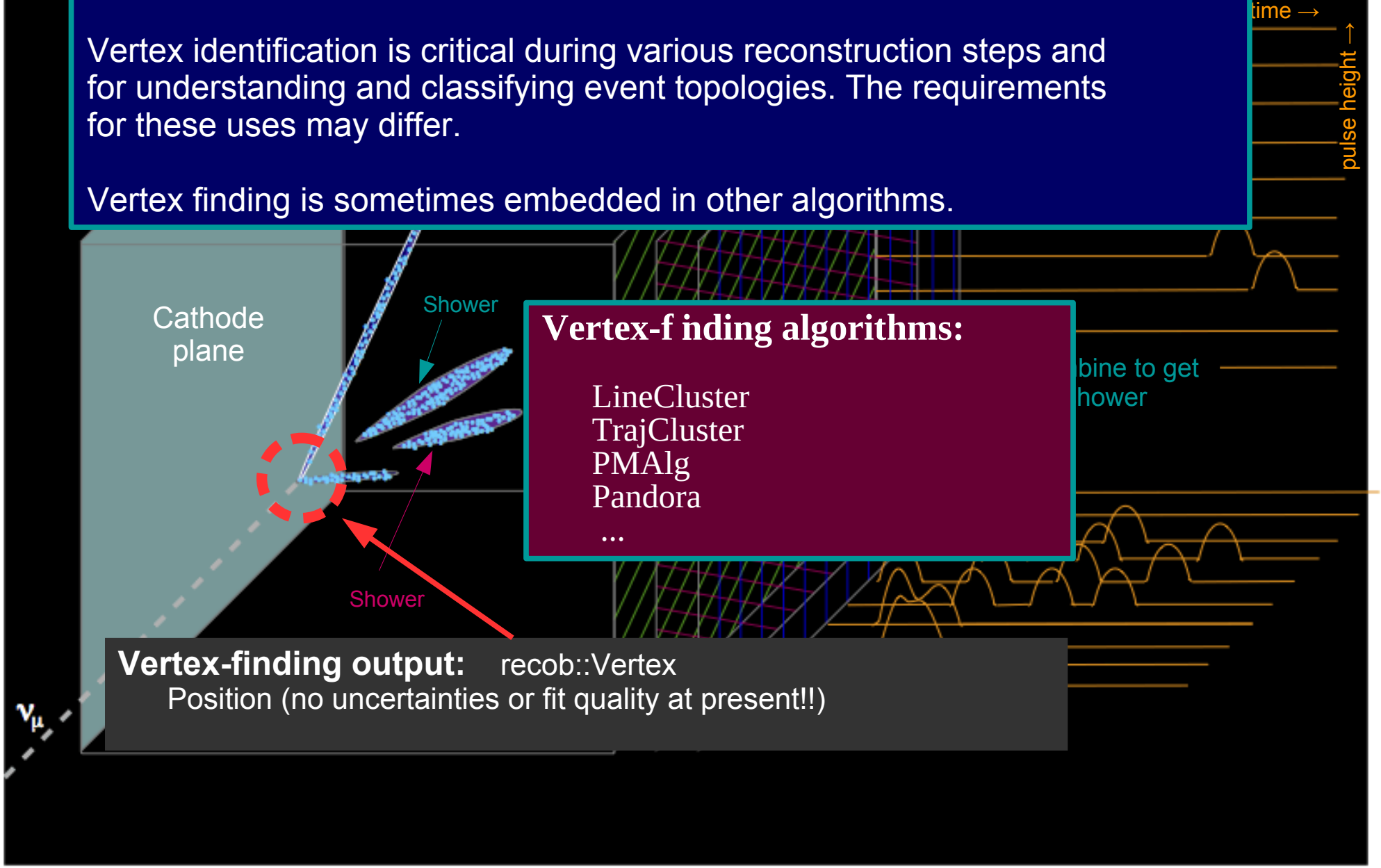
Vertex-finding output: recob::Vertex
Position (no uncertainties or fit quality at present!!)

Reconstruction workflow and data structures

4) Identify vertices in the event from tracks in the event

Vertex identification is critical during various reconstruction steps and for understanding and classifying event topologies. The requirements for these uses may differ.

Vertex finding is sometimes embedded in other algorithms.

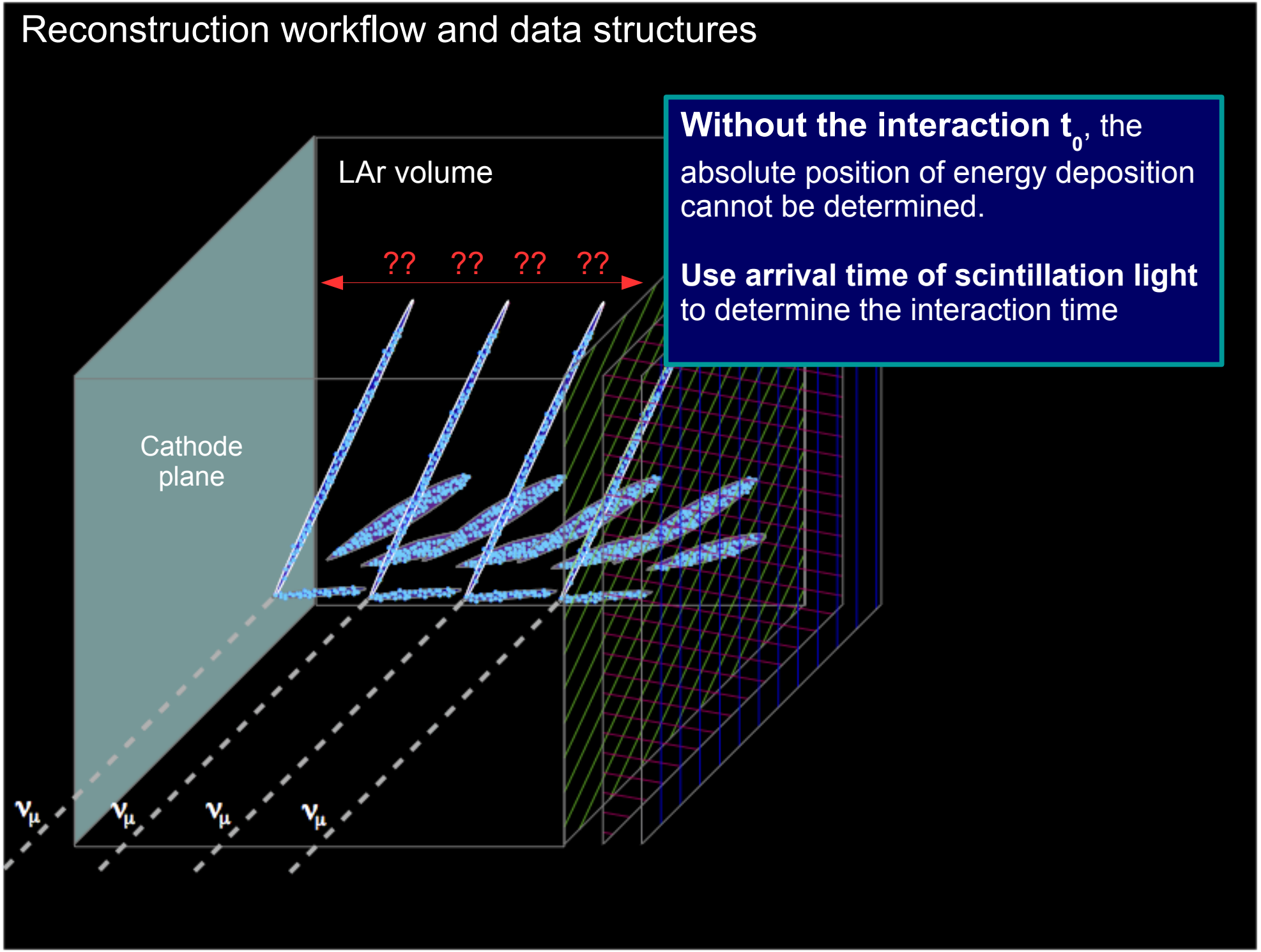


Vertex-f nding algorithms:

LineCluster
TrajCluster
PMAlg
Pandora
...

Vertex-finding output: recob::Vertex
Position (no uncertainties or fit quality at present!!)

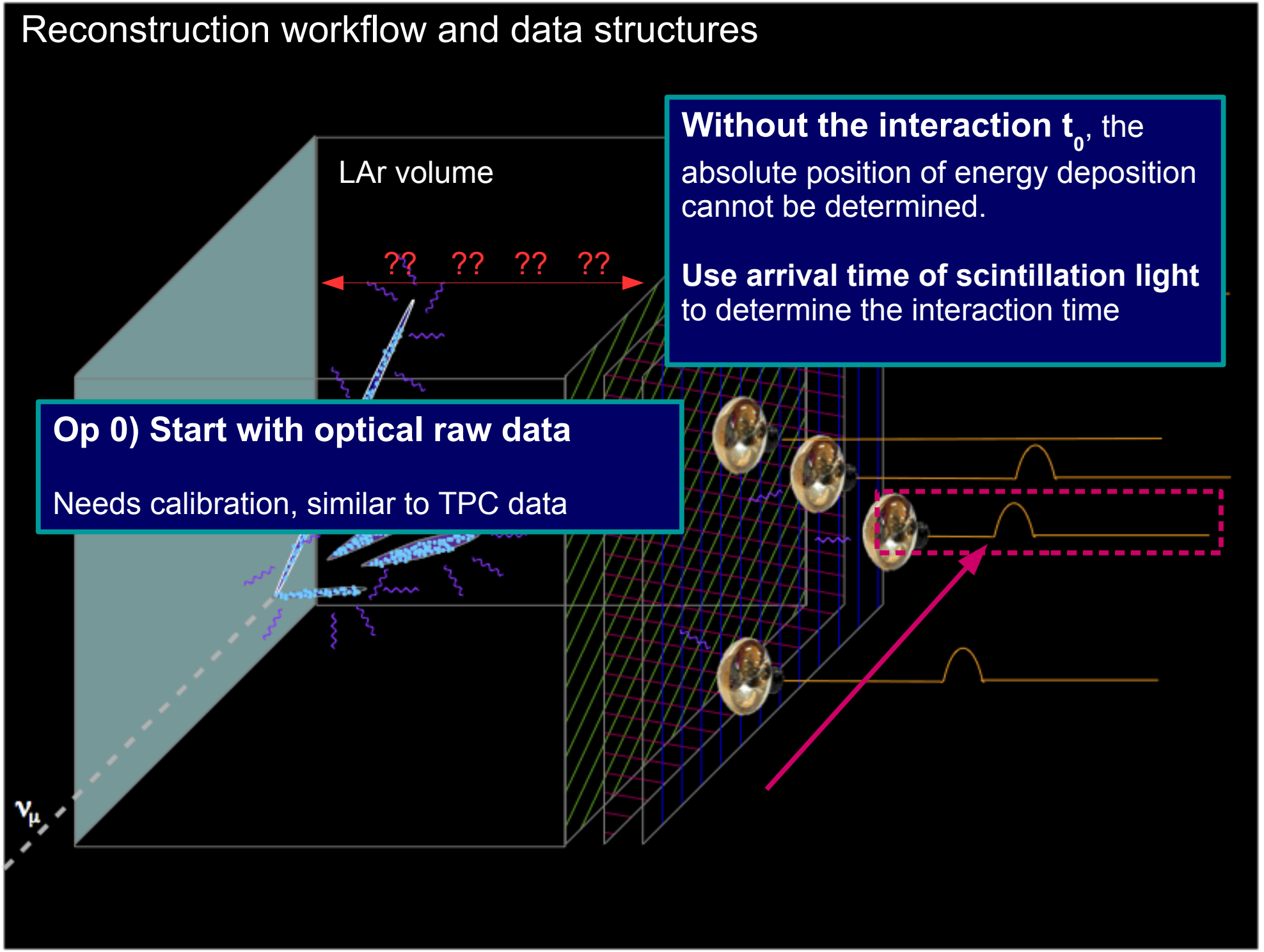
Reconstruction workflow and data structures



Without the interaction t_0, the absolute position of energy deposition cannot be determined.

Use arrival time of scintillation light to determine the interaction time

Reconstruction workflow and data structures



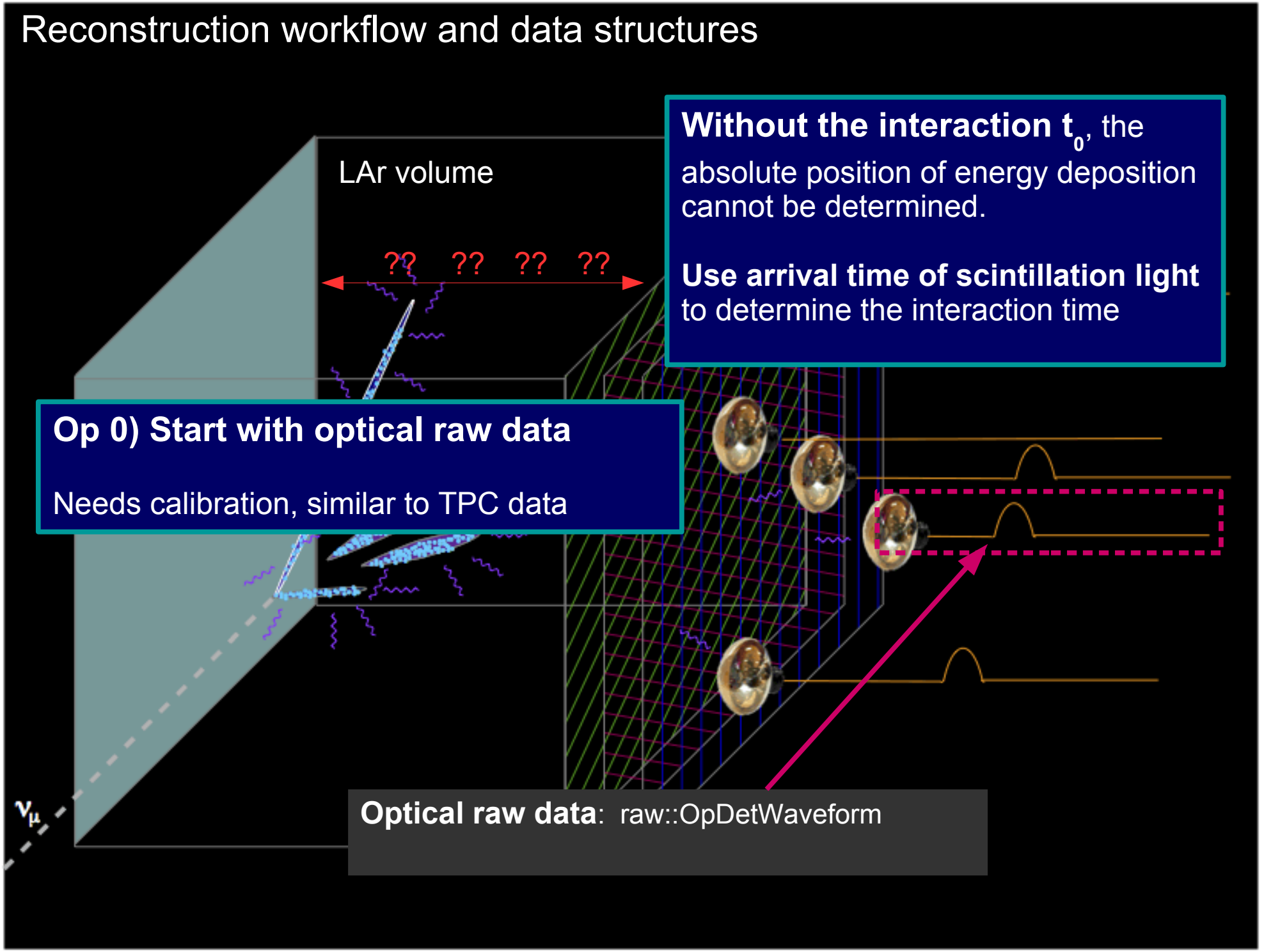
Without the interaction t_0, the absolute position of energy deposition cannot be determined.

Use arrival time of scintillation light to determine the interaction time

Op 0) Start with optical raw data

Needs calibration, similar to TPC data

Reconstruction workflow and data structures



Without the interaction t_0, the absolute position of energy deposition cannot be determined.

Use arrival time of scintillation light to determine the interaction time

Op 0) Start with optical raw data

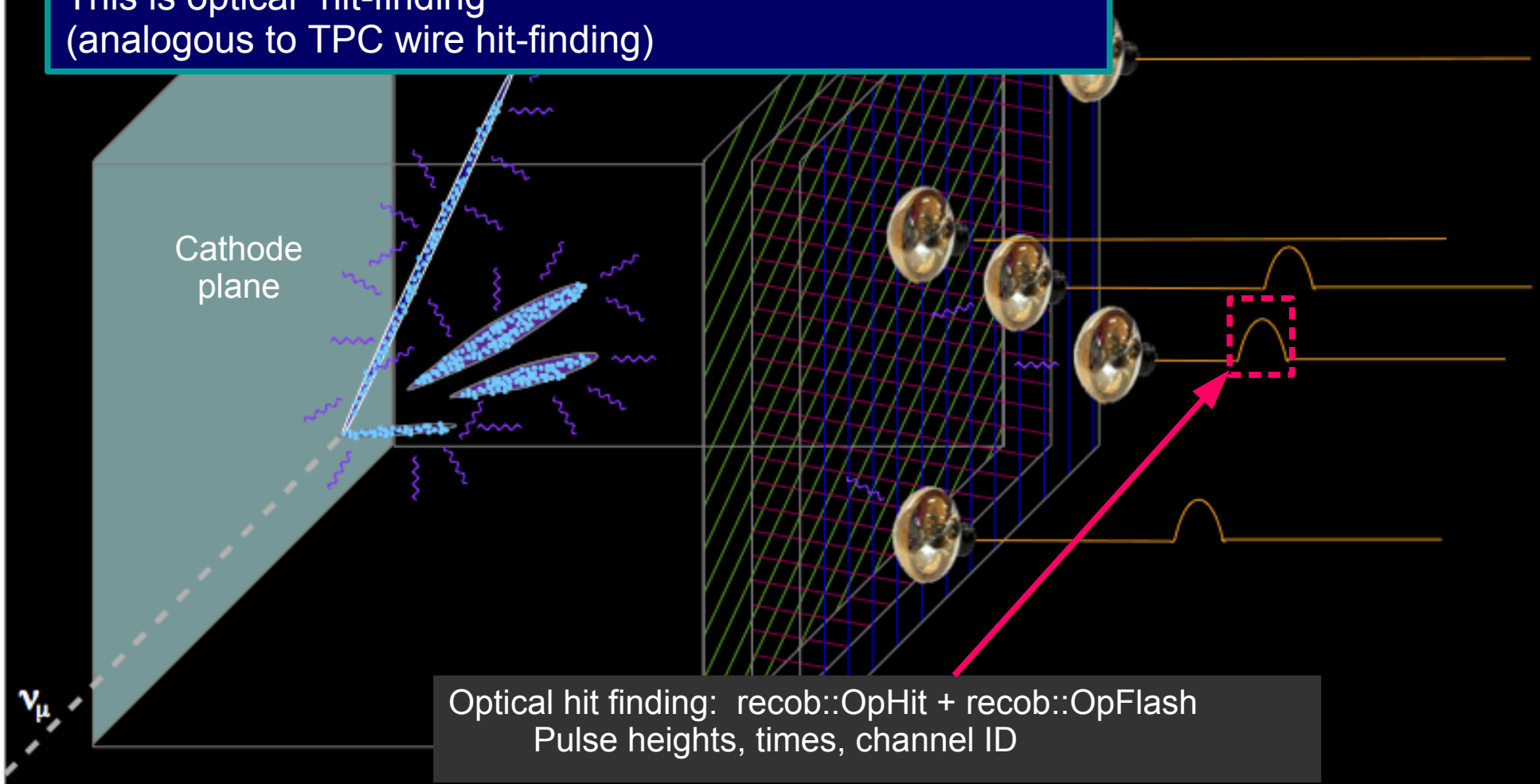
Needs calibration, similar to TPC data

Optical raw data: raw::OpDetWaveform

Reconstruction workflow and data structures

Op 1) First identify signals from single interactions
on each optical channel

This is optical “hit-finding”
(analogous to TPC wire hit-finding)

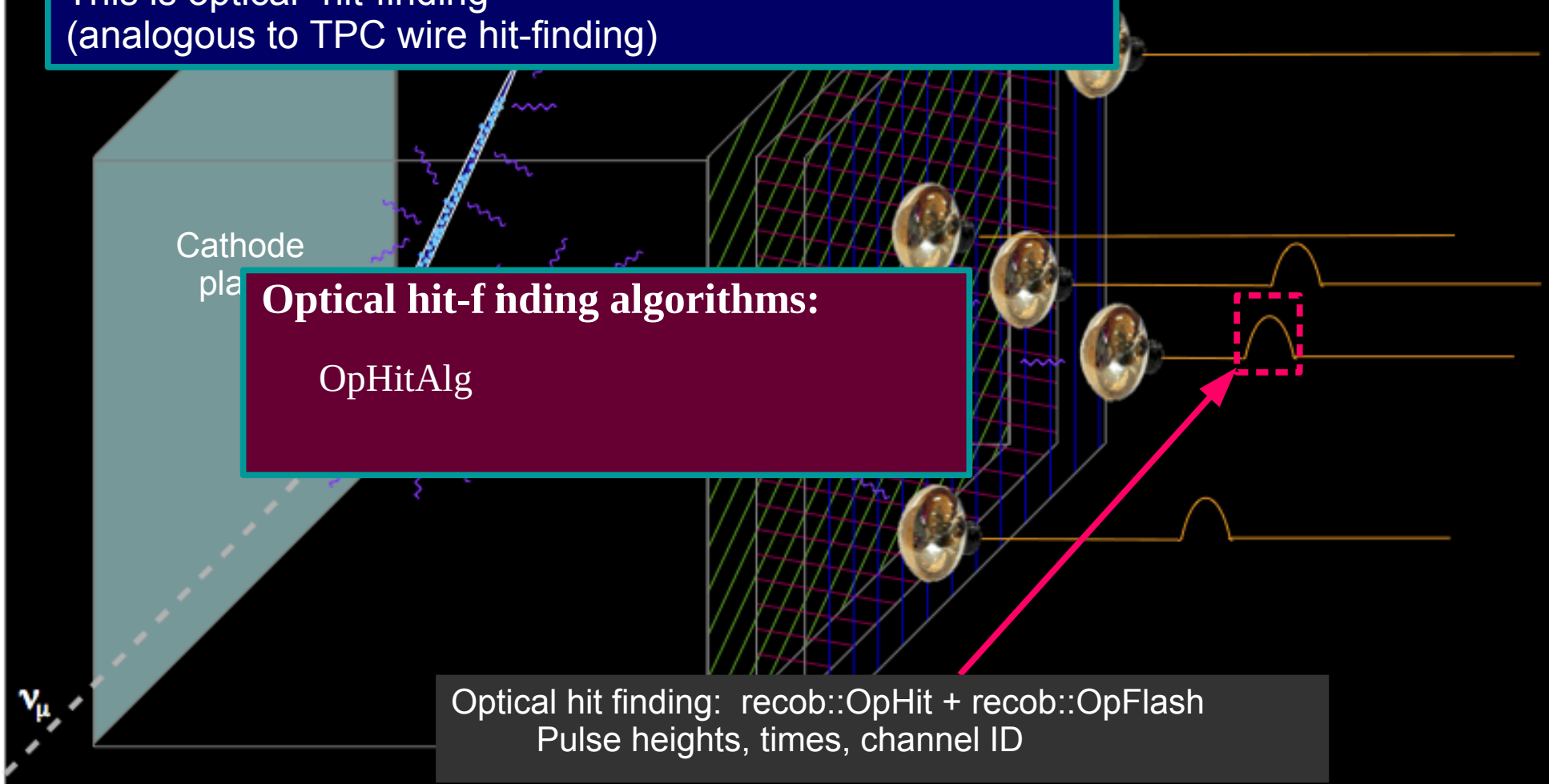


Optical hit finding: recob::OpHit + recob::OpFlash
Pulse heights, times, channel ID

Reconstruction workflow and data structures

Op 1) First identify signals from single interactions
on each optical channel

This is optical “hit-finding”
(analogous to TPC wire hit-finding)



Optical hit-f nding algorithms:

OpHitAlg

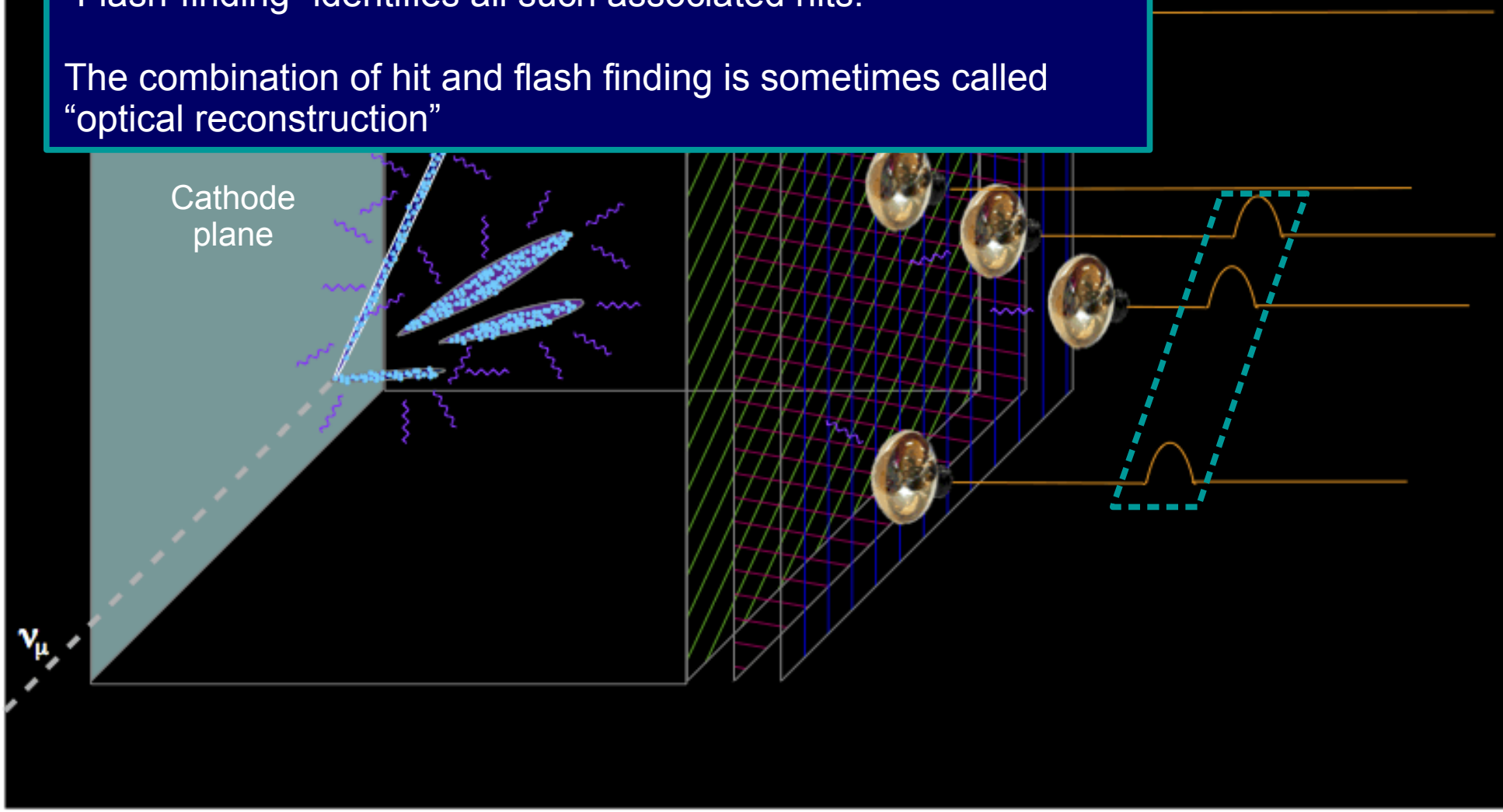
Optical hit finding: recob::OpHit + recob::OpFlash
Pulse heights, times, channel ID

Reconstruction workflow and data structures

Op 2) Find all hits associated with a single interaction in the TPC.

All tracks, showers from a single interaction produce a “flash”. “Flash-finding” identifies all such associated hits.

The combination of hit and flash finding is sometimes called “optical reconstruction”

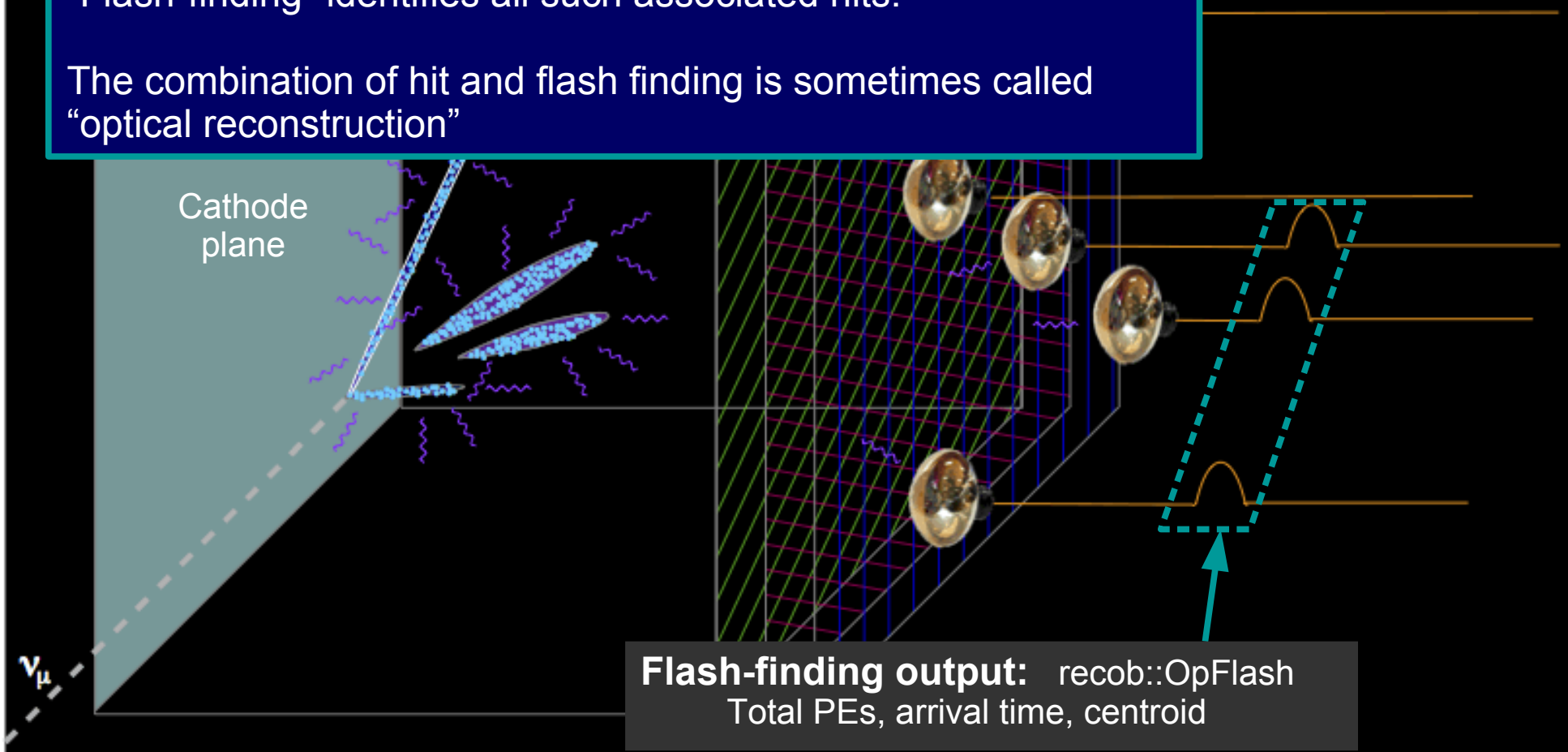


Reconstruction workflow and data structures

Op 2) Find all hits associated with a single interaction in the TPC.

All tracks, showers from a single interaction produce a “flash”. “Flash-finding” identifies all such associated hits.

The combination of hit and flash finding is sometimes called “optical reconstruction”

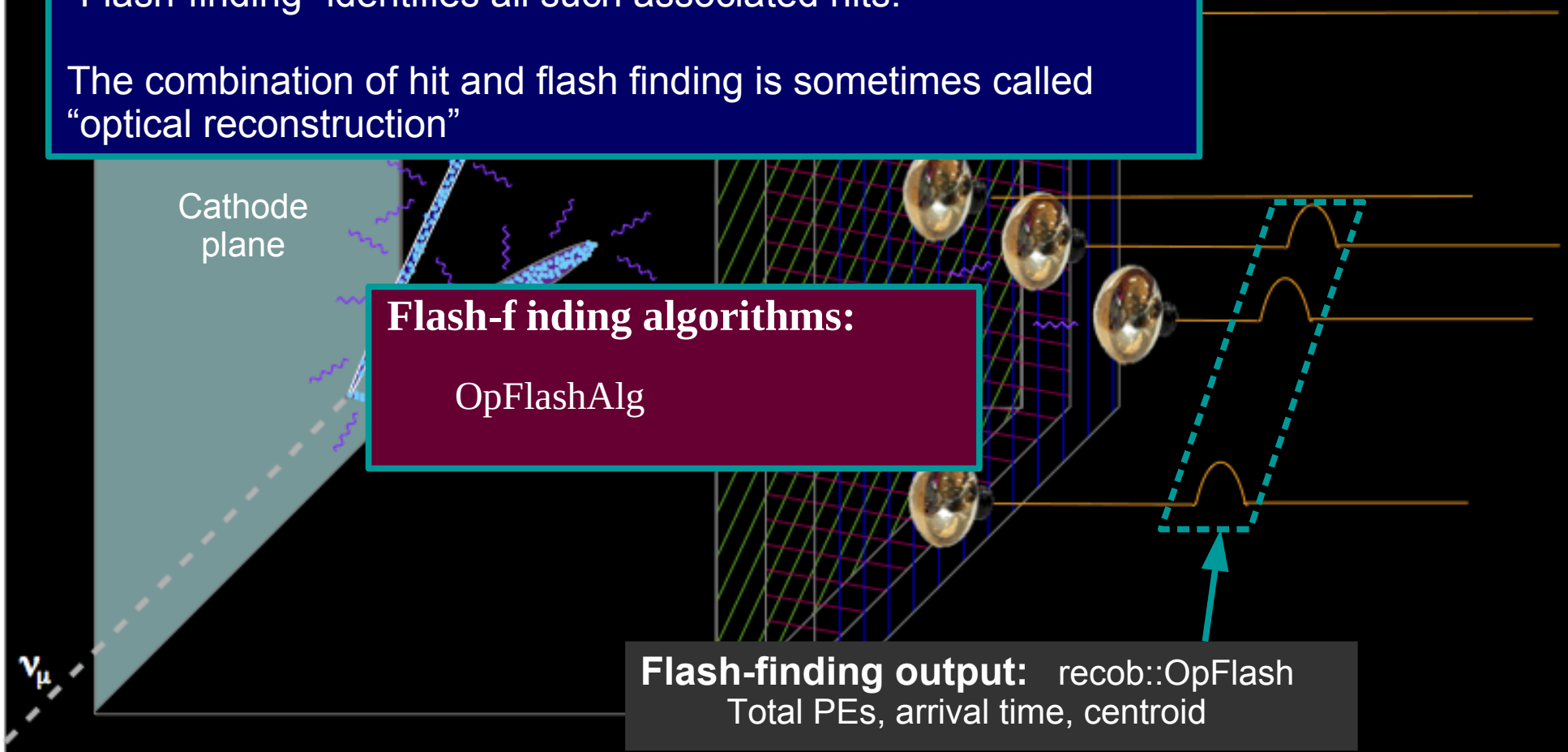


Reconstruction workflow and data structures

Op 2) Find all hits associated with a single interaction in the TPC.

All tracks, showers from a single interaction produce a “flash”. “Flash-finding” identifies all such associated hits.

The combination of hit and flash finding is sometimes called “optical reconstruction”



Flash-f nding algorithms:

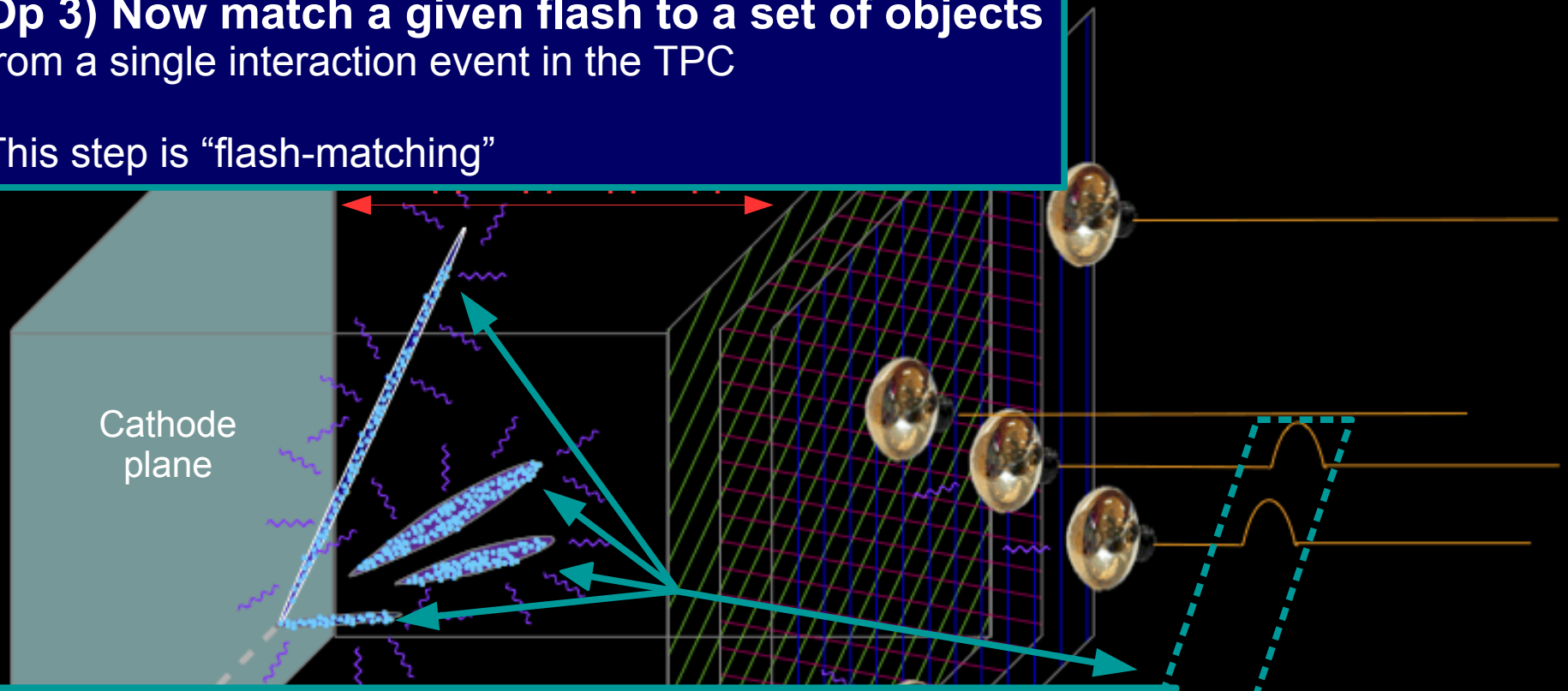
OpFlashAlg

Flash-finding output: recob::OpFlash
Total PEs, arrival time, centroid

Reconstruction workflow and data structures

Op 3) Now match a given flash to a set of objects from a single interaction event in the TPC

This step is “flash-matching”



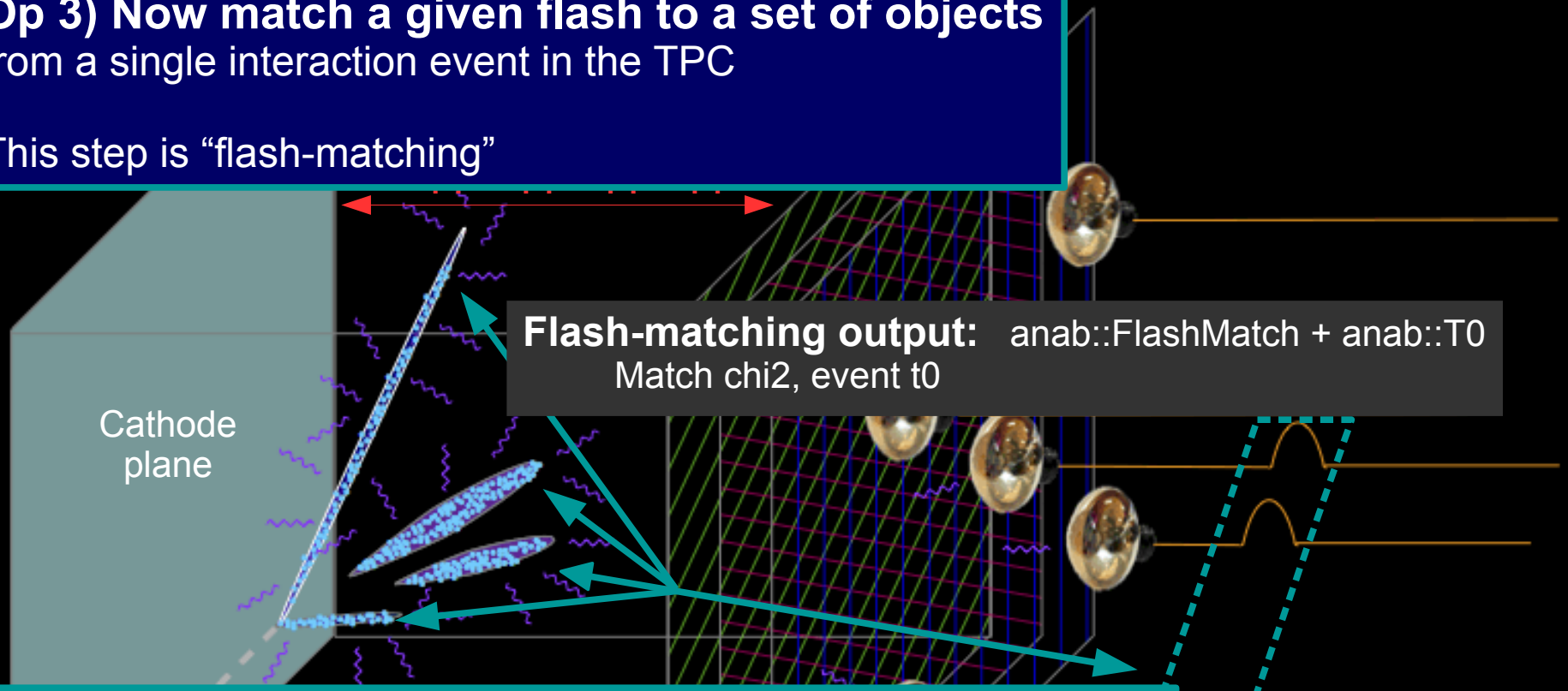
Requires **comparing observed signal pattern with predicted**, so need to know all the objects that were in the interaction **Estimate event t0** from this process.

Because it requires an event hypothesis, **may be performed at analysis-level** rather than at primary reconstruction

Reconstruction workflow and data structures

Op 3) Now match a given flash to a set of objects from a single interaction event in the TPC

This step is “flash-matching”



Flash-matching output: anab::FlashMatch + anab::T0
Match chi2, event t0

Requires **comparing observed signal pattern with predicted**, so need to know all the objects that were in the interaction **Estimate event t0** from this process.

Because it requires an event hypothesis, **may be performed at analysis-level** rather than at primary reconstruction

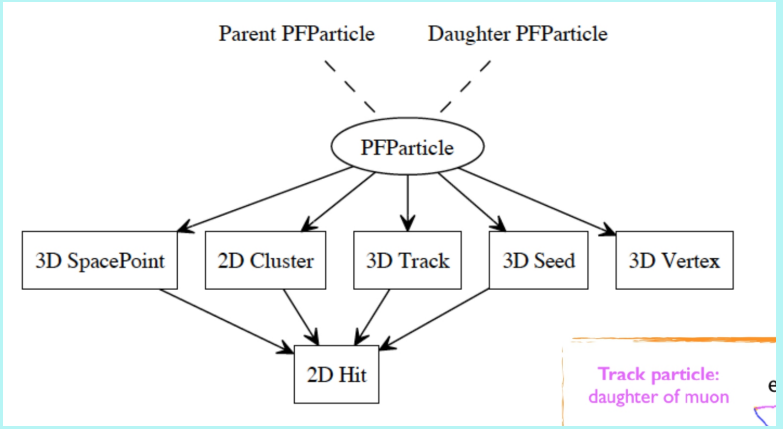
Special algorithm notes

Pandora

- Multi-algorithm pattern recognition framework

  See Lorena Escuerdo's talk on Pandora from Wednesday

  - Takes recob::Hit collections as input, and performs
    - 2D cluster finding (track-like and shower-like)
    - 3D matching
    - shower/track discrimination
    - vertex finding and classification
  - Produces recob::PFParticles
    - Mother-daughter hierarchy for all particles from a single event vertex
      - Distinguisb tracks, delta rays, showers, vertices



Pandora

- Multi-algorithm pattern recognition framework

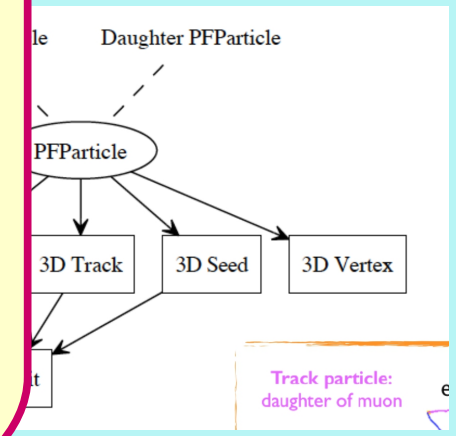
  See Lorena Escuerdo's talk on Pandora from Wednesday

  - Takes rec
    - 2D cl
    - 3D m
    - show
    - verte
  - Produces
    - Moth from
      - Dis vertices

Developed externally to LArSoft

- Dedicated, direct interface modules in LArSoft
- Typically run in stages with intervening LArSoft algorithms

  e.g., Pandora cosmic tagging
  → LArSoft cosmic hit removal
  → Pandora neutrino event reco

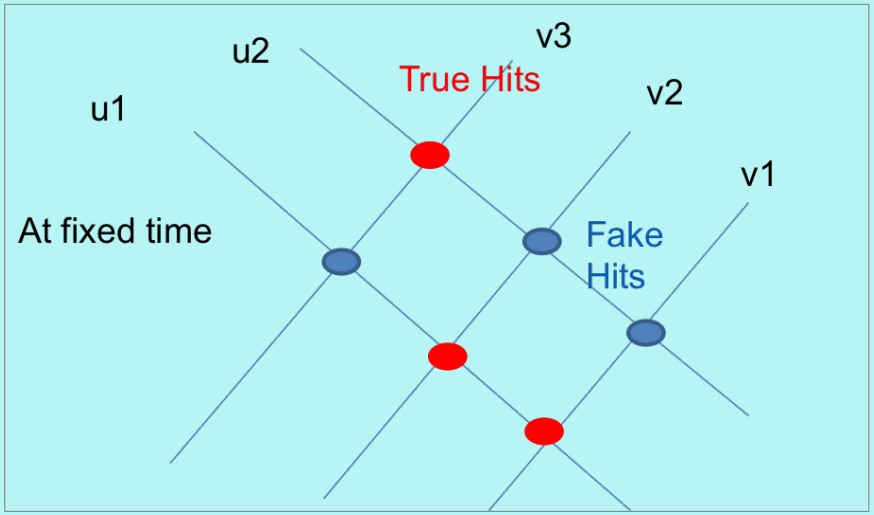


Wire-cell

- 3D object reconstruction from 2D time slices

  See http://www.phy.bnl.gov/wire-cell/

- Uses charge as constraint to assign hits values at each wire crossing
  - Demanding signal processing requirements
  - Computationally challenging
  - High reward



- Developed externally to LArSoft
  - Direct, multipoint interfacing to LArSoft in progress
  - May also drive data product evolution

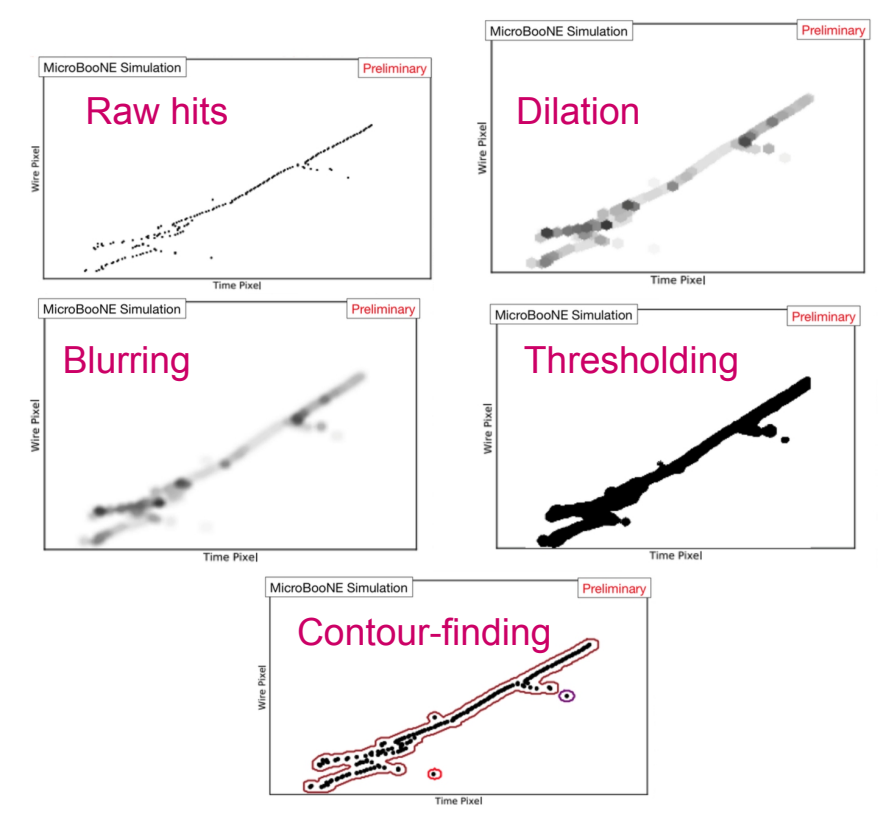
Deep learning networks

- A variety of efforts are in progress
  - At present, aimed mainly at event classification, region of interest identification, track/shower discrimination
- One already integrated into LArSoft

  See talk by P. Plonski, D. Stefan, R. Sulej on Wednesday
  - Provide hit-level shower/track discrimination and vertex identification upstream of conventional LArSoft algorithms
- Much to learn about how to use these types of algorithms
  - Envision continuing integration work

Image processing techniques

- MicroBooNE effort underway to perform 2D clustering with image processing software
  - Topology-based clustering
  - OpenCV product applied to 2D TPC image data
  - interesting early results
- Future work
  - How to use this technique
  - How to integrate into LArSoft



From MicroBooNE-Note-1012-Pub

Other techniques...

...that I've not mentioned, or am not aware of

Still lots of room for innovation, evolution, so be bold!!

Additional reconstruction considerations

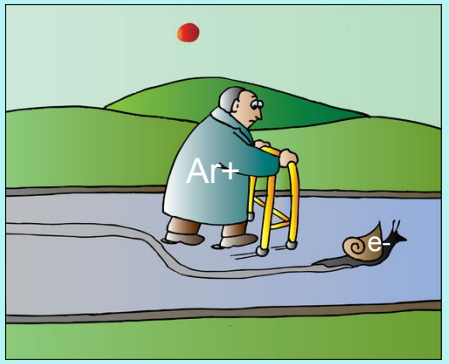
Additional reconstruction phases

- Cosmic ray removal
  - Particularly important for surface detectors
    - SBN detectors at Fermilab
    - Test beam detectors
  - Current algorithms primarily geometry-based
    - Look for out of time tracks, or in-time tracks that cross a boundary
    - CR hits can be removed for downstream clustering / tracking / shower-finding
  - Representative algorithms:
    - CosmicTrackTagger
    - CosmicPFParticleTagger
  - Output: anab::CosmicTag

Additional reconstruction phases

- Non-shower calorimetric measurements
  - Energy and dE/dx estimates for Tracks
  - Representative algorithms: CalorimetryAlg, TrackCalorimetryAlg
  - Output: anab::Calorimetry
- Momentum estimation and particle identification
  - Use range, dE/dx and multiple Coulomb scattering of tracks
  - Representative algorithms:
    - Track3DKalmanHit, TrajCluster yield MCS momentum
    - Chi2PIDAlg, PIDAAlg perform particle ID
  - Output: anab::ParticleID, Assns<Track, ParticlePID>, or TTree

Other complications

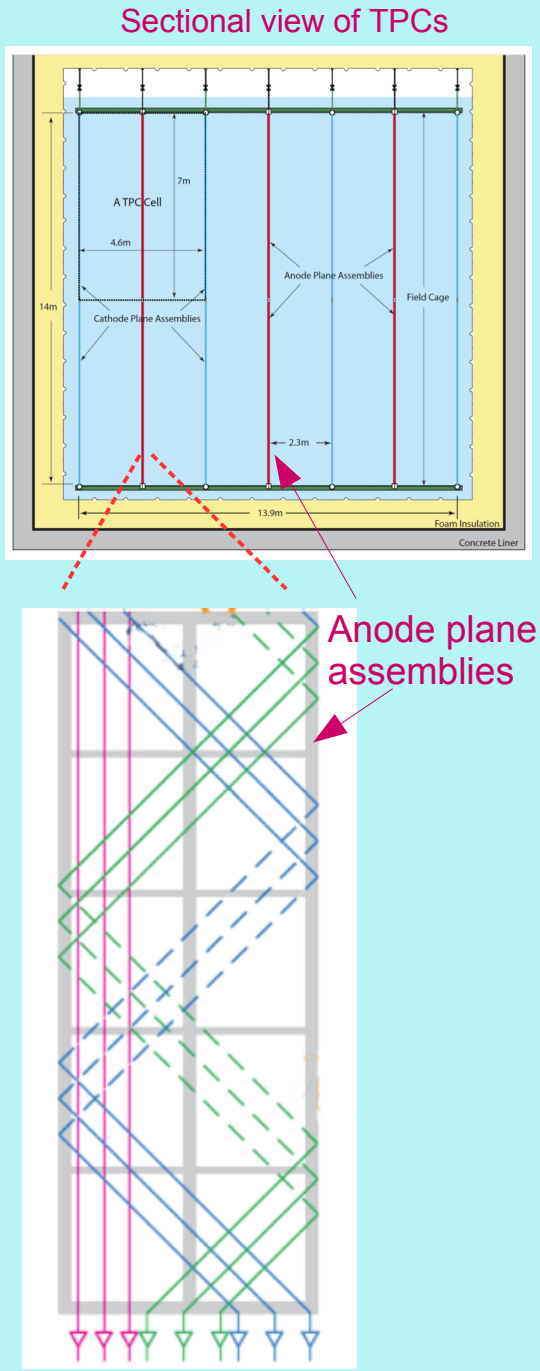


Alexei Talimonov, used w/o permission

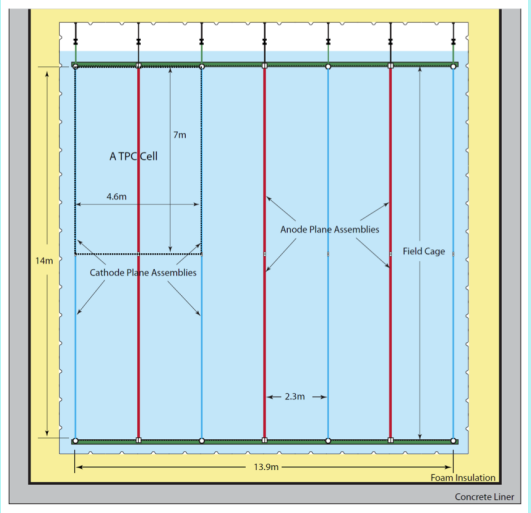
- Space-charge distortions
  - Ion drift mobilities about $10^6\times$ smaller than for electrons
    - Cation drift velocities are ~nm / μs !!
  - **For surface detectors**, cosmic rays introduces large (+) ion load
    - At **ProtoDUNE SP**, electron drift distortions reach **few 10's cm scale**
    - Need to map and correct for these field distortions
  - A common service interface exists to access the offsets
    - Allows experiment-dependent implementations
- Charge attenuation
  - Electron lifetime can be comparable to maximum drift time
  - Charge yield at wires will be drift-length dependent
  - Affects S:N ratio and charge / energy measurements

Other complications

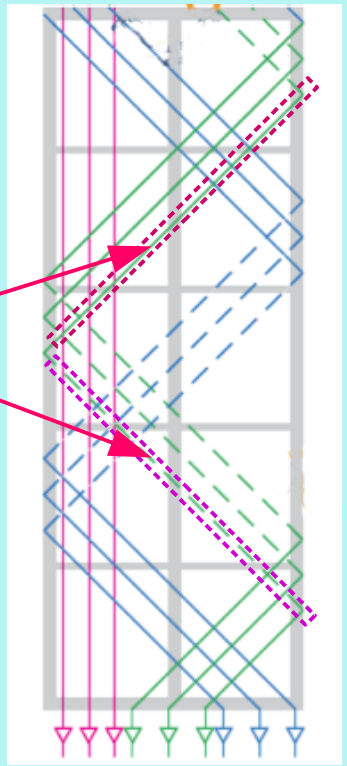
- Hit disambiguation
  - DUNE TPCs have wrapped induction wires
    - See signals in two TPCs, and in many cases, multiple places within the same TPC
  - Some LArSoft nomenclature:
    - “Wire”: a segment of a physical wire in an anode plane
      - A geometric concept
    - “Channel”: a readout channel connected to one or more “wires”
      - A DAQ concept



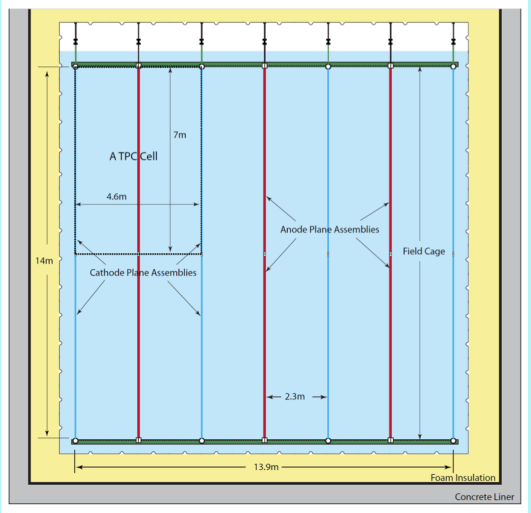
Other complications



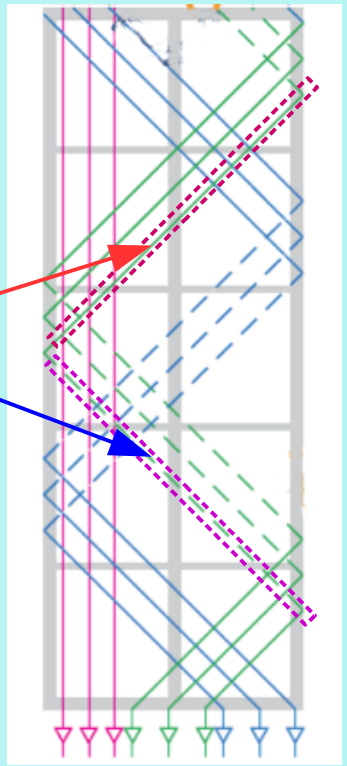
- Hit disambiguation
  - DUNE TPCs have wrapped induction wires
    - See signals in two TPCs, and in many cases, multiple places within the same TPC
  - Some LArSoft nomenclature:
    - "Wire": a segment of a physical wire in an anode plane
      - A geometric concept
    - "Channel": a readout channel connected to one or more "wires"
      - A DAQ concept



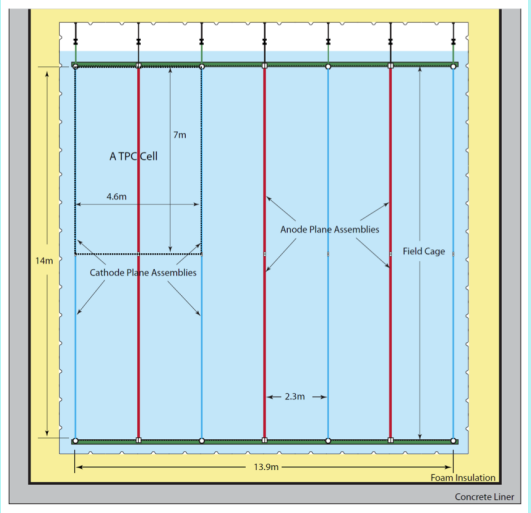
Other complications



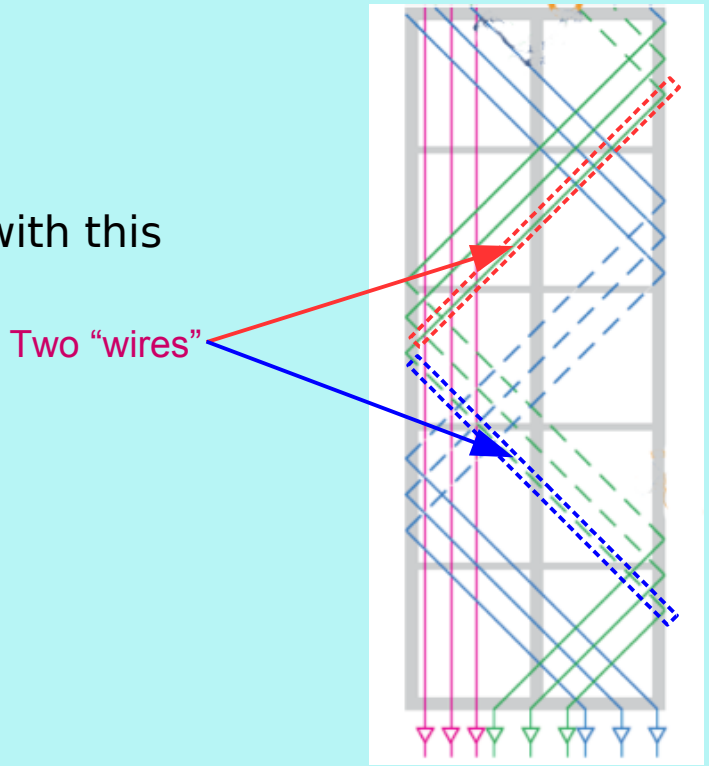
- Hit disambiguation
  - DUNE TPCs have wrapped induction wires
    - See signals in two TPCs, and in many cases, multiple places within the same TPC
  - Some LArSoft nomenclature:
    - “Wire”: a segment of a physical wire in an anode plane
      - A geometric concept
    - “Channel”: a readout channel connected to one or more “wires”
      - A DAQ concept



Other complications



- Hit disambiguation
  - DUNE TPCs have wrapped induction wires
    - See signals in two TPCs, and in many cases, multiple places within the same TPC
  - Introduce a disambiguation step to deal with this
    - Resolves the TPC ambiguity of each induction hit
    - Currently performed after hit-finding
      - Existing algorithms use timing information and neighboring activity



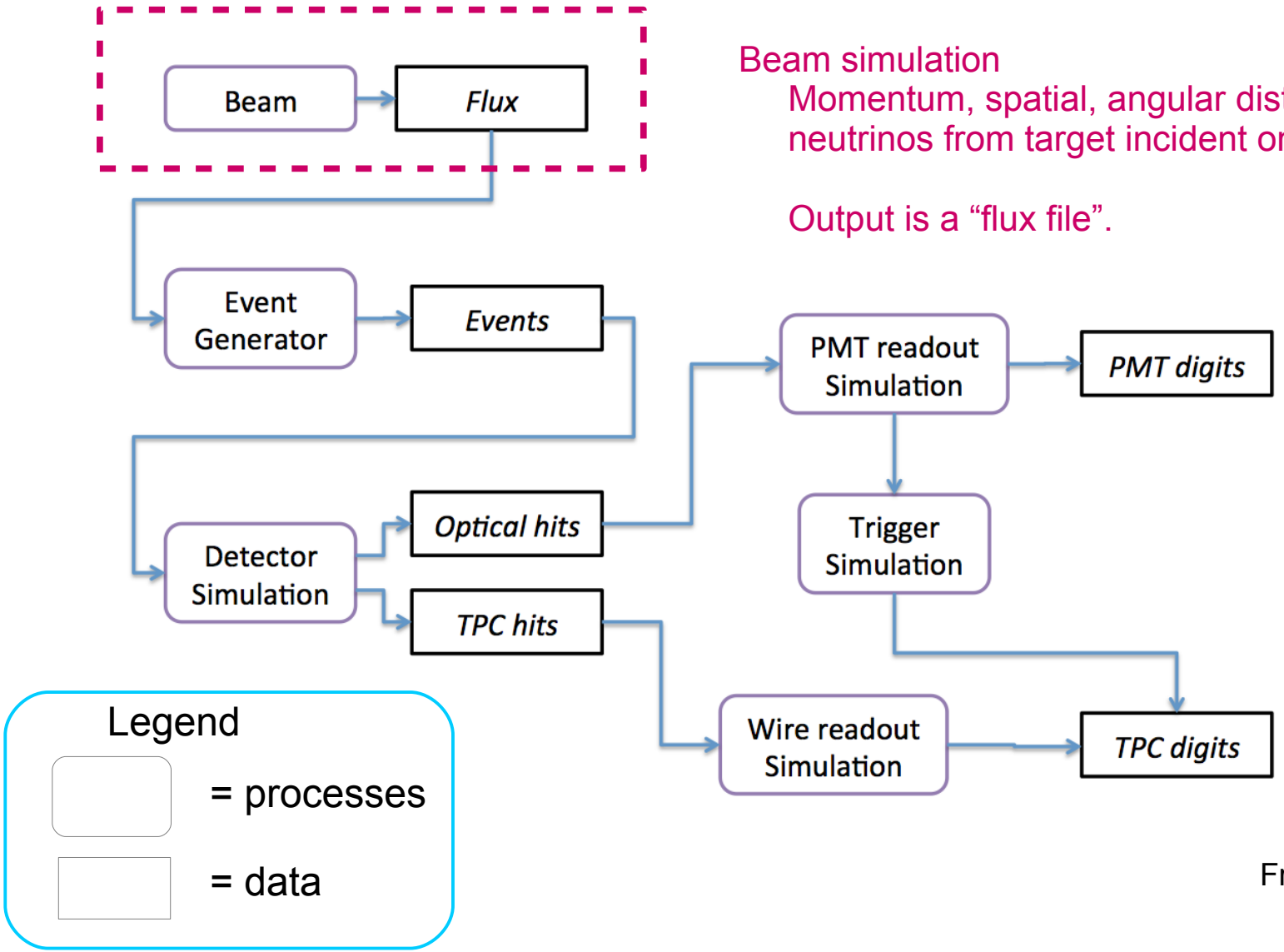
Dual phase TPC

- ProtoDUNE DP
  - Significant progress over the past year
    - See talk by D. Stefan, R. Sulej from Wednesday
  - A basic simulation and reconstruction workflow runs
    - Required some code changes, parameter tuning
    - But still using core, common LArSoft algorithms
  - Need to try other algorithms, workflows
  - A long list of improvements and change suggested

Detector simulation in LArSoft

Simulation workflow

Three phases, typically run separately



Beam simulation

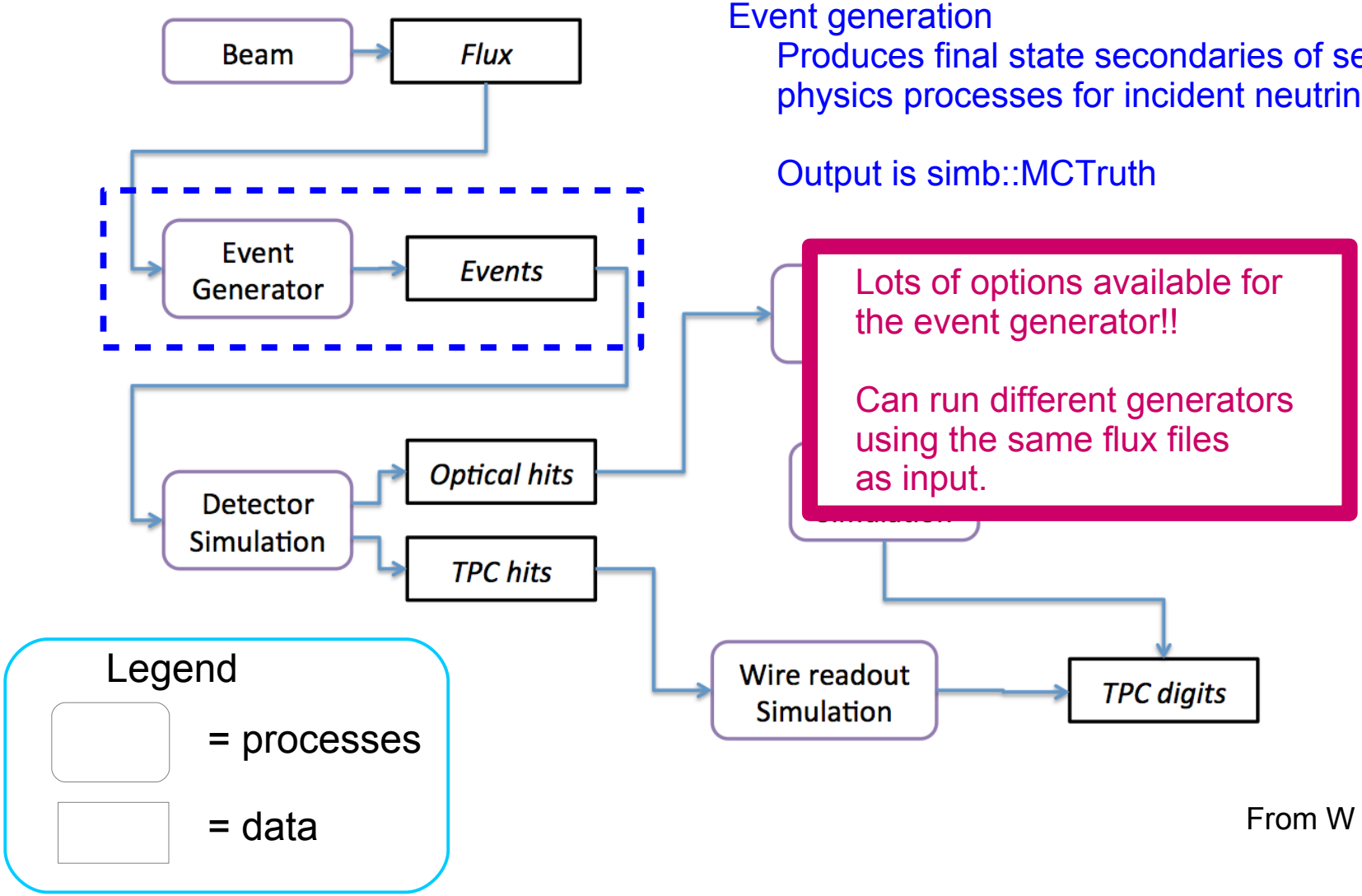
Momentum, spatial, angular distribution of neutrinos from target incident on detector

Output is a “flux file”.

From W Seligman

Simulation workflow

Three phases, typically run separately



Event generation

Produces final state secondaries of selected physics processes for incident neutrinos.

Output is simb::MCTruth

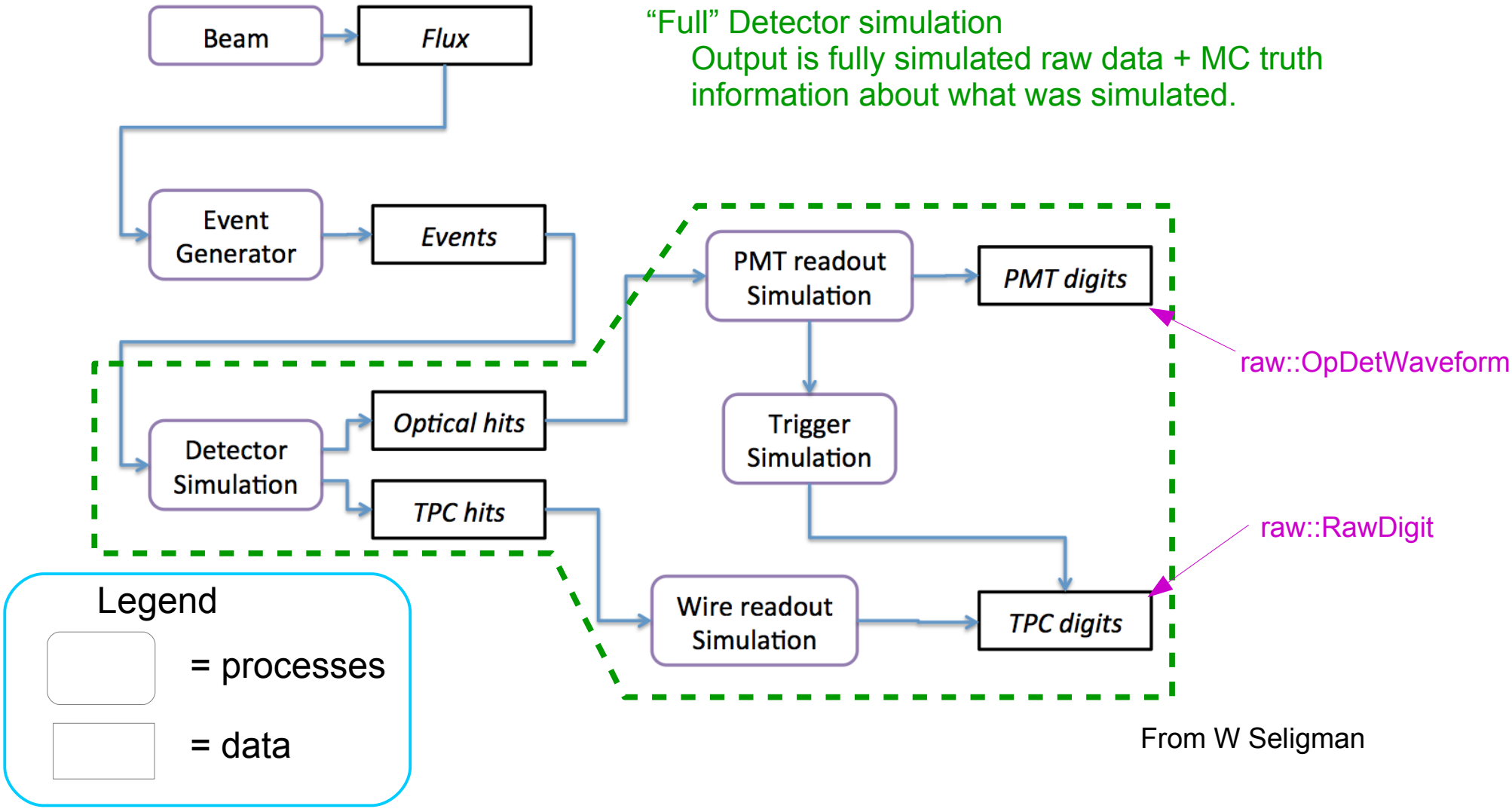
Lots of options available for the event generator!!

Can run different generators using the same flux files as input.

From W Seligman

Simulation workflow

Three phases, typically run separately

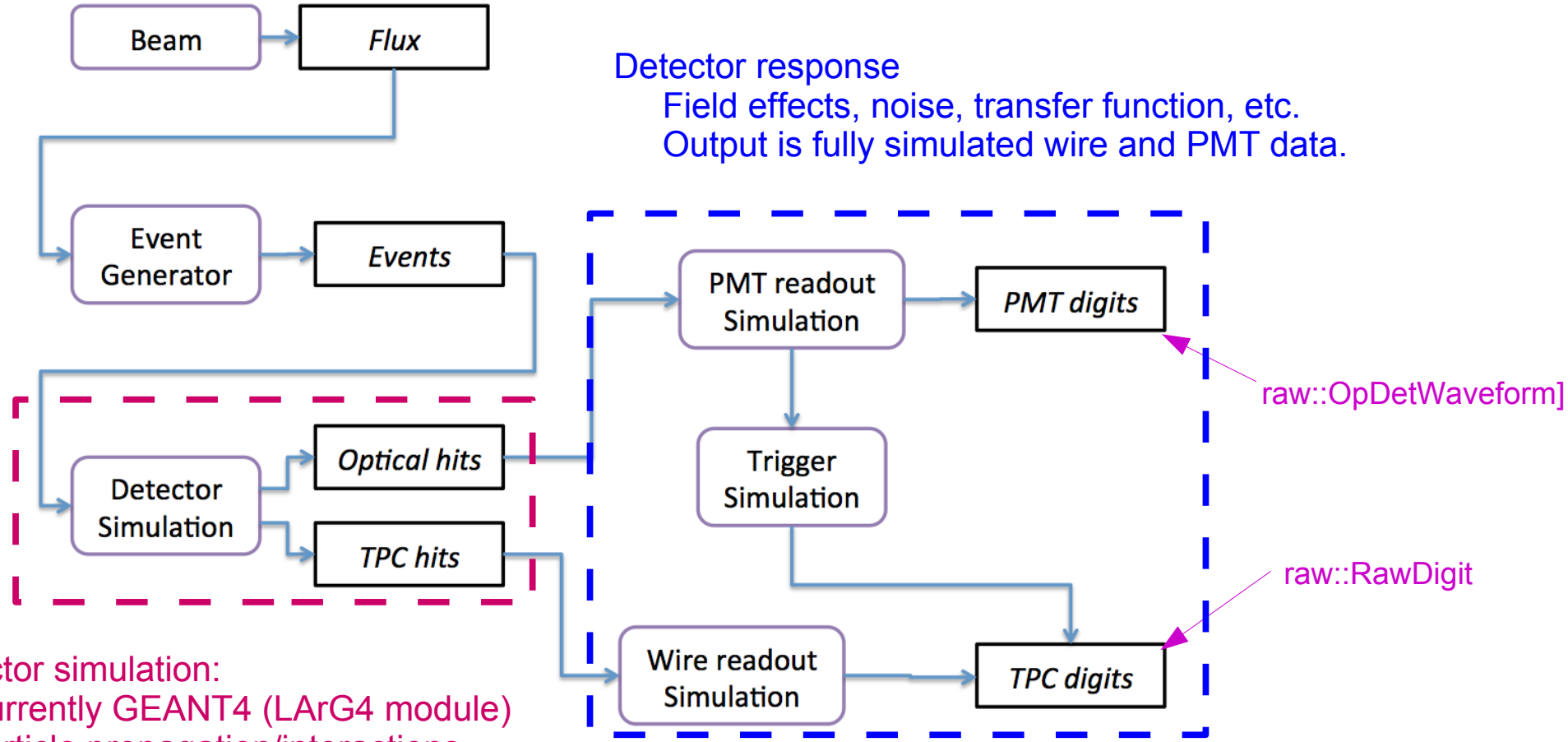


"Full" Detector simulation
Output is fully simulated raw data + MC truth information about what was simulated.

From W Seligman

Simulation workflow

The full detector simulation includes two separable sub-phases



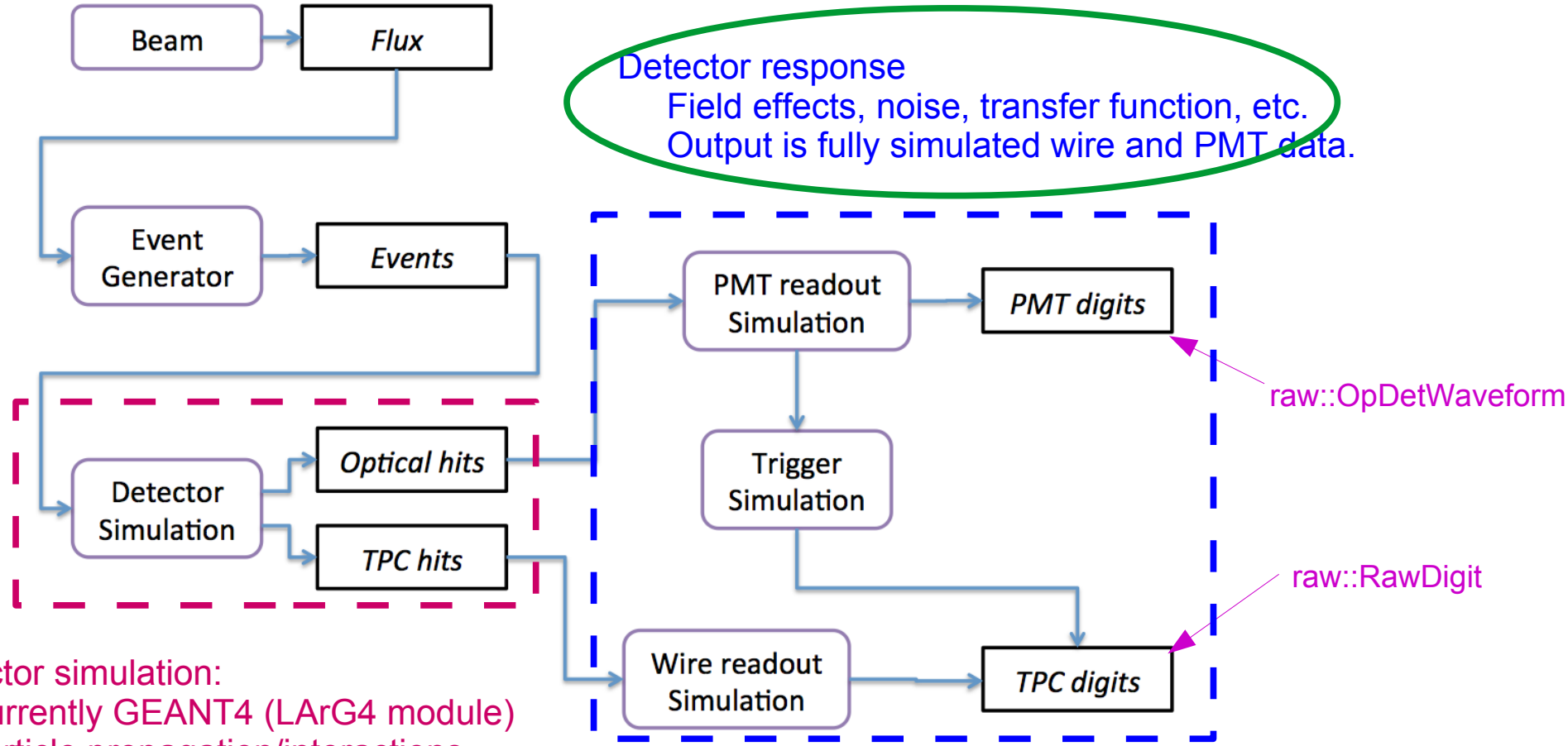
Detector response
Field effects, noise, transfer function, etc.
Output is fully simulated wire and PMT data.

Detector simulation:
Currently GEANT4 (LArG4 module)
Particle propagation/interactions, charge/photon transport.
Output is MC truth information in simb::MCParticle, sim::SimPhoton, sim::SimChannel, sim::AuxDetSimChannel,

From W Seligman

Simulation workflow

Most detect-specific customizations go into the detector response



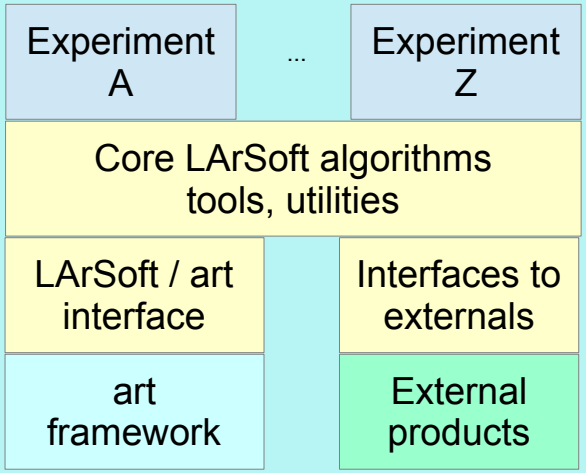
Detector simulation:
Currently GEANT4 (LArG4 module)
Particle propagation/interactions, charge/photon transport.
Output is MC truth information in simb::MCParticle, sim::SimPhoton, sim::SimChannel, sim::AuxDetSimChannel,

From W Seligman

LArSoft design principles

LArSoft design principles and practices

- Detector interoperability
  - The most important design objective for the LArSoft project
  - Define (and use!!) common interfaces for accessing detector-specific configuration information and functionality
  - Applies to geometry, channel mappings, LAr properties, E-field map...
- Separation of framework and algorithm code
  - Encapsulate algorithms, configuration, tools and utilities into a layer that is independent of the art framework
  - Many benefits follow from this



General disclaimer:

In examining the code, you may note that only a portion currently adheres to these principles.

- An on-going architecture review project is intended to address this

Strongly encourage people to adopt these practices for all new code

LArSoft design principles and practices

- Use of standardized algorithm interfaces
  - Define standard interfaces for well-defined steps in the workflow to promote modularity, layering of algorithms
- Modularity
  - Build sophistication by applying algorithms in a layered, iterative structure
- Design / write testable units of code
  - Include unit and integration testing in the development process
  - Follow the practice of continuous integration
    - Perform automated, broad-scale testing at frequent intervals in order to catch unintended side-effects quickly

LArSoft design principles and practices

- Continuous integration
  - Automated tests run every time code is pushed to central git repositories
    - Code authors need to provide the relevant tests!!
  - See http://larsoft.org/continuous-integration
- Document code in the source files as it is written
  - See many files with **no comments at all** describing what the code does
  - At very least, need the have at the top of all header files:
    - the purpose of the file / code
    - pre-requisites and assumptions
    - Anything else people **need to know** in order to use it
- Document algorithms and services on http://larsoft.org/add
  - High-level description, the principal author, etc.

Using LArSoft

Supported platforms

See https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/_Supported_platforms_

- Scientific Linux
  - SLF6 (the reference system) + SLF7
    - Should work on any SL variant; Works on SLC6 (CERN), Redhat 6 (SLAC)
- Mac OSX
  - Mavericks and Yosemite
    - ups qualifiers d13:noifdh and d14:noifdh respectively
  - “Known to work” on El Capitan
    - Must disable SIP and install openssl
- Ubuntu
  - “Known to work” with Ubuntu14, demonstrated for 15 and 16
    - LArSoft team distributes installation tarballs for u14 (best effort only)

Installation instructions:

See links in release notes available at
https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/LArSoft_release_list

LArSoft code repositories

- Code lives in a set of git repositories hosted at Fermilab

larcore	Low level utilities
larcoreobj	Low level data products
larcorealg	Low level utilities
lardata	Data products
lardataobj	Data products
lardataalg	Low level algorithms
larevt	Low level algorithms that use data products
larsim	Simulation code
larreco	Primary reconstruction code
larana	Secondary reconstruction and analysis code
lareventdisplay	LArSoft-based event display
larpandora	LArSoft interface to Pandora
larexamples	Placeholder for examples

Repositories shareable with art-independent frameworks. List can be expanded.

LArSoft code repositories

- Code lives in a set of git repositories hosted at Fermilab

larcore	Low level utilities
larcoreobj	Low level data products
larevt	Low level algorithms that use data products
larsim	Simulation code
larreco	Primary reconstruction code
larana	Secondary reconstruction and analysis code
lareventdisplay	LArSoft-based event display
larpandora	LArSoft interface to Pandora
larexamples	Placeholder for examples

1) All publicly accessible at http://cdcvs.fnal.gov/projects/<repository name>

2) For read/write access: ssh://p-<repository name>@cdcvs.fnal.gov/cvs/projects/<repository name>> (requires valid kerberos ticket)

Repositories shareable with art-independent frameworks. List can be expanded.

LArSoft products

- The build procedure creates and installs a **ups product** from the code in each repository

larcore	Low level utilities
larcoreobj	Low level data products
larcorealg	Low level utilities
lardata	Data products
lardataobj	Data products
lardataalg	Low level algorithms
larevt	Low level algorithms that use data products
larsim	Simulation code
larreco	Primary reconstruction code
larana	Secondary reconstruction and analysis code
lareventdisplay	LArSoft-based event display
larpandora	LArSoft interface to Pandora
larexamples	Placeholder for examples

Each product is self-contained, aside from dependencies

LArSoft releases

- A LArSoft release is a consistent set of LArSoft products built from tagged versions of code in the repositories
  - Implicitly includes corresponding versions of all external dependencies used to build it
  - Each release of LArSoft has a release notes page on scisoft
    - http://scisoft.fnal.gov/scisoft/bundles/larsoft/<version>/larsoft-<version>.html
- larsoft
  - A larsoft umbrella product binds it all together to give it one version, one setup command
    - setup larsoft v06_06_00 -q …
- larsoft_data
  - A ups product for large configuration files

larsoft v04.16.00	
Product	**Version**
larcore	v04.13.00
lardata	v04.11.00
larevt	v04.08.06
larsim	v04.08.03
larreco	v04.12.00
larana	v04.08.00
lareventdisplay	v04.06.00
larpandora	v04.04.16
larexamples	v04.04.16
larsoft_data	v0.04.00

…

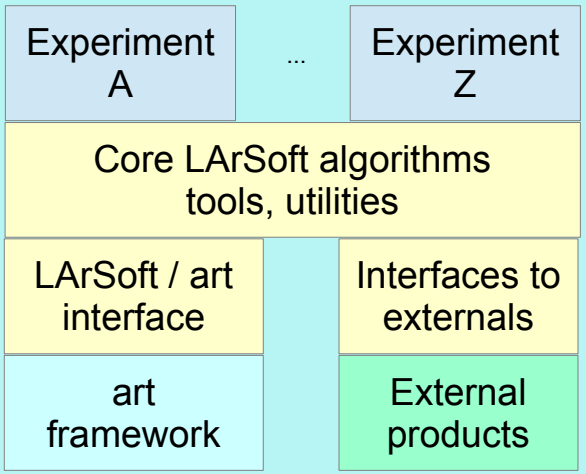
LArSoft releases

- Two types of releases
  - Integration
    - Created weekly or on demand for special purposes
    - Contents approved at Coordination Meetings
      - Head of develop + additional branches approved at a CM or via email
    - May be removed without notice after about a month
      - In practice, we announce our intentions in advance
  - Production
    - Any release designated as “production” by an experiment
    - Created on demand (but usually on the weekly schedule)
    - Contents approved by the experiment declaring production
      - Typically also coordinated through the CM to keep other experiments informed
    - Production releases are retained on disk indefinitely
- List of all available tagged releases
  - https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/LArSoft_release_list

How to set up and run art/LArSoft

- First point to note
  - LArSoft is designed to be run by experiments
    - Need detector-specific parts to run it
    - So start with the code of your experiment

DUNE	dunetpc
LArIAT	lariatsoft
MicroBooNE	uboonecode
SBND	sbndcode



The setup procedures for each are different so refer to the relevant setup instructions

  - For DUNE, https://cdcvs.fnal.gov/redmine/projects/dunetpc/wiki/_Tutorial_

- Second point to note
  - Do not need a “working area” to run LArSoft. Just need to set up the appropriate products + a fcl file

How to set up and run art/LArSoft

- The most simple scenario: run from a tagged LArSoft release
  - First, set up the working environment:

```
# Set up ups
source <ups location>/setup
#
# Set up the working environment
# for your experiment
<the setup procedure for your experiment here>

# Set up a LArSoft release
setup larsoft v06_06_00 -q e10:<prof|debug>
```

    - In most cases, experiment-specific setup scripts will include all of the above
    - Note that setting up for development requires additional steps
  - Now run art (the LArSoft version is called `lar`)

```
# List art command-line options
lar --help
#
# Now run LArSoft
#
lar -c <some fcl file>.fcl [-i <input file>]
```

That's it! ...if you have a fcl file and have no code to change

Setting up and running LArSoft as a developer

Use this basic procedure to modify code, including your own art-based analysis code

Start with the same setup procedure from the previous page

Note that mrb is the primary build tool used by LArSoft

mrb –help to list commands
mrb <command> --help for more information

```
# Create working area
mkdir workdir
cd workdir
mrb newDev -v v06_06_00 -q e10:prof

# Now perform local setup
source localProducts_larsoft_v06_06_00_e10_prof/setup

# Move to source area and check out code
cd srcs
mrb g <repository name>   # or mrb gitCheckout ...

# Develop code
cd <repository name>/...

# Set up build environment
cd $MRB_BUILDDIR
mrbsetenv

# Build and install ('install' will actually do both)
mrb b -j<N>           # or mrb build; N = # cores to use
mrb I -j <N>          # or mrb install

# Set up newly built code
mrbslp                      # 'slp' = setup local products

# Run as before
lar -c ...
```

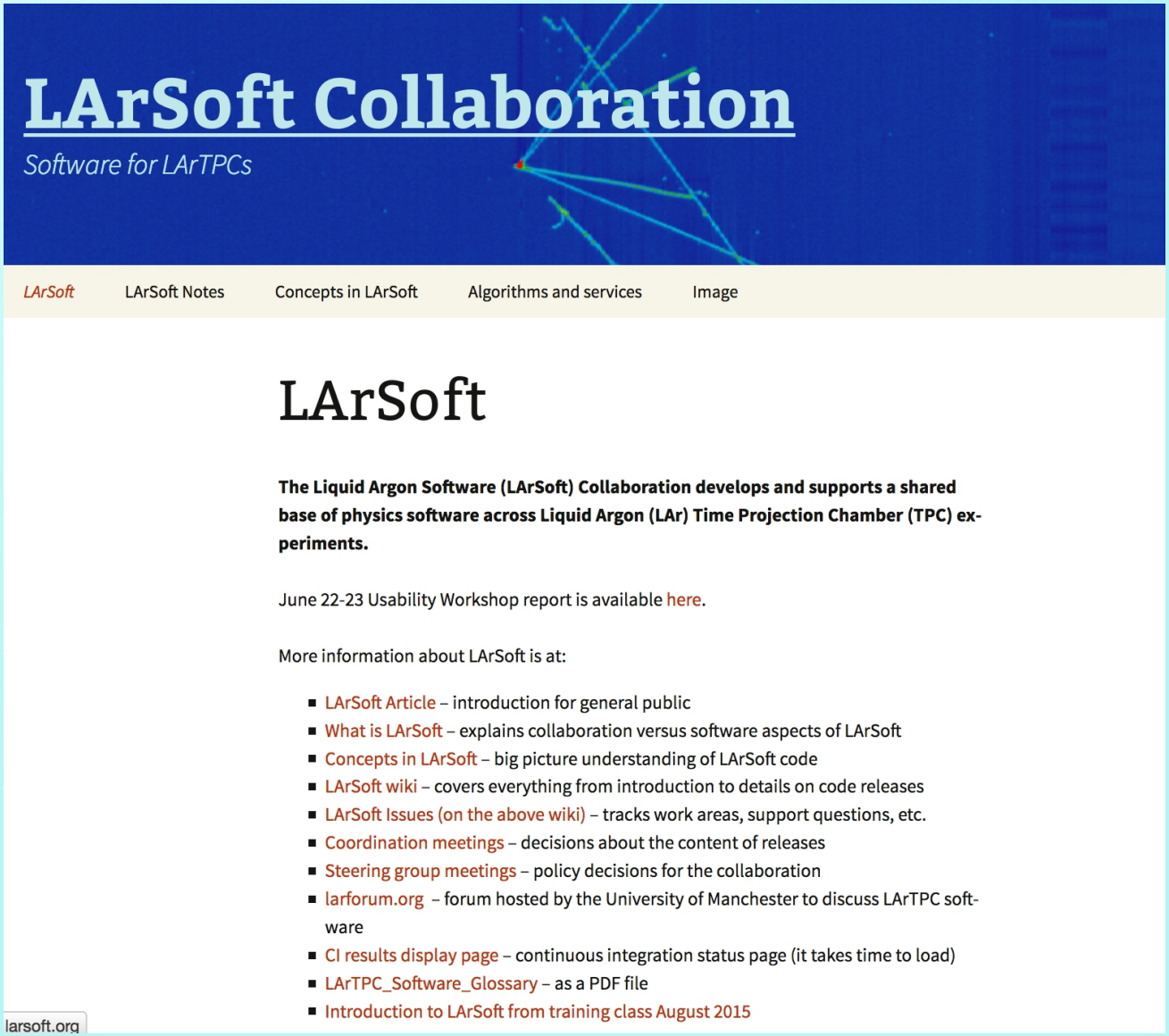
Getting LArSoft to work with a new detector

- LArSoft is really a toolkit
  - It requires a certain amount of detector-specific information and plug-in functionality in order to work
- The minimum needed to run LArSoft for a new detector
  - Define the geometry for the new detector in a GDML file
  - Customize E-field, drift velocity, readout parameters, etc, as needed
  - Customize digitization for simulation, as needed
  - Write a fcl file
  - lar -c sim_new_det.fcl → simulated data; lar -c <reco...>.fcl → results!!

  ...A bit over-simplified, but this is basically what happens

Resources

larsoft.org



The public facing home page for the LArSoft Collaboration

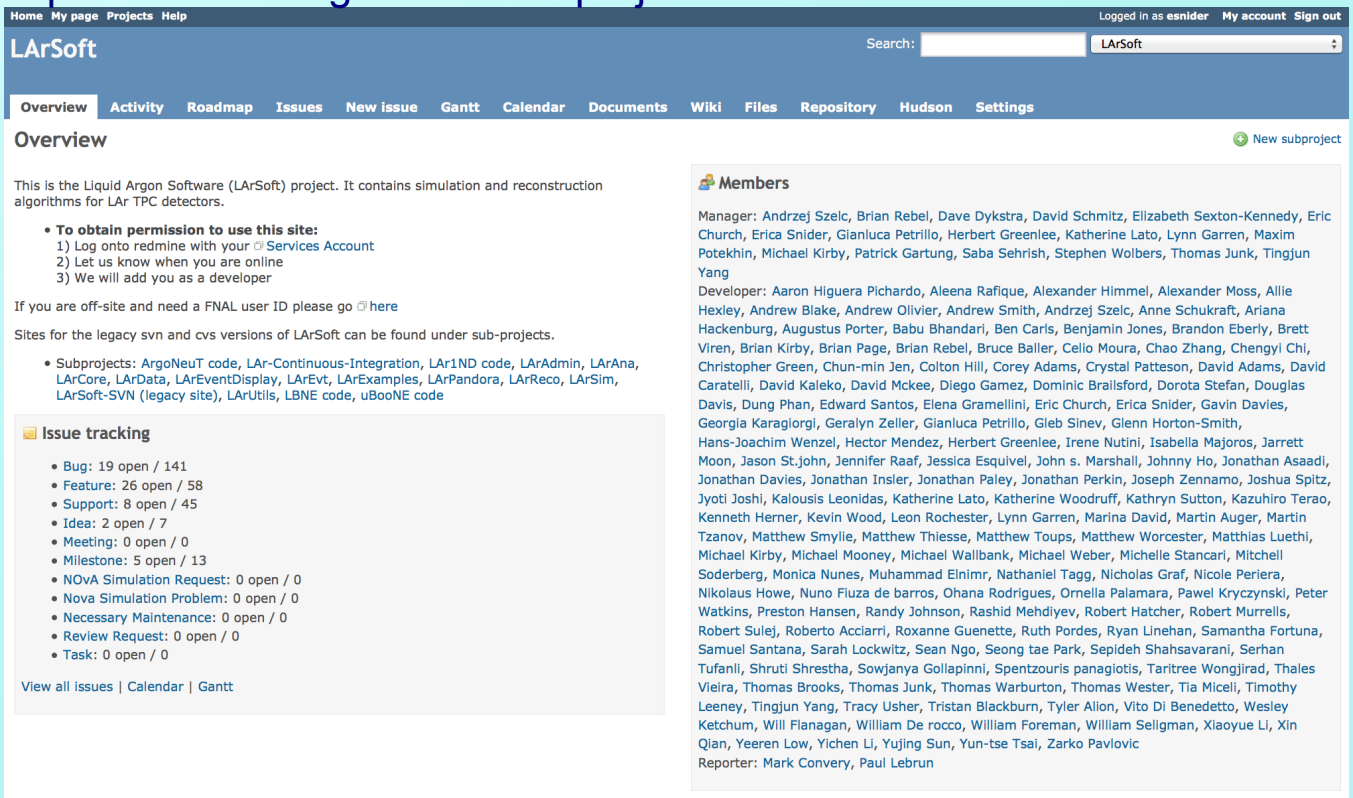
Contains introductory information + links to further information and resources

Tabs across the top link to different types of content

LArSoft Redmine site

Redmine sites are called “projects”

https://cdcvs.fnal.gov/redmine/projects/larsoft



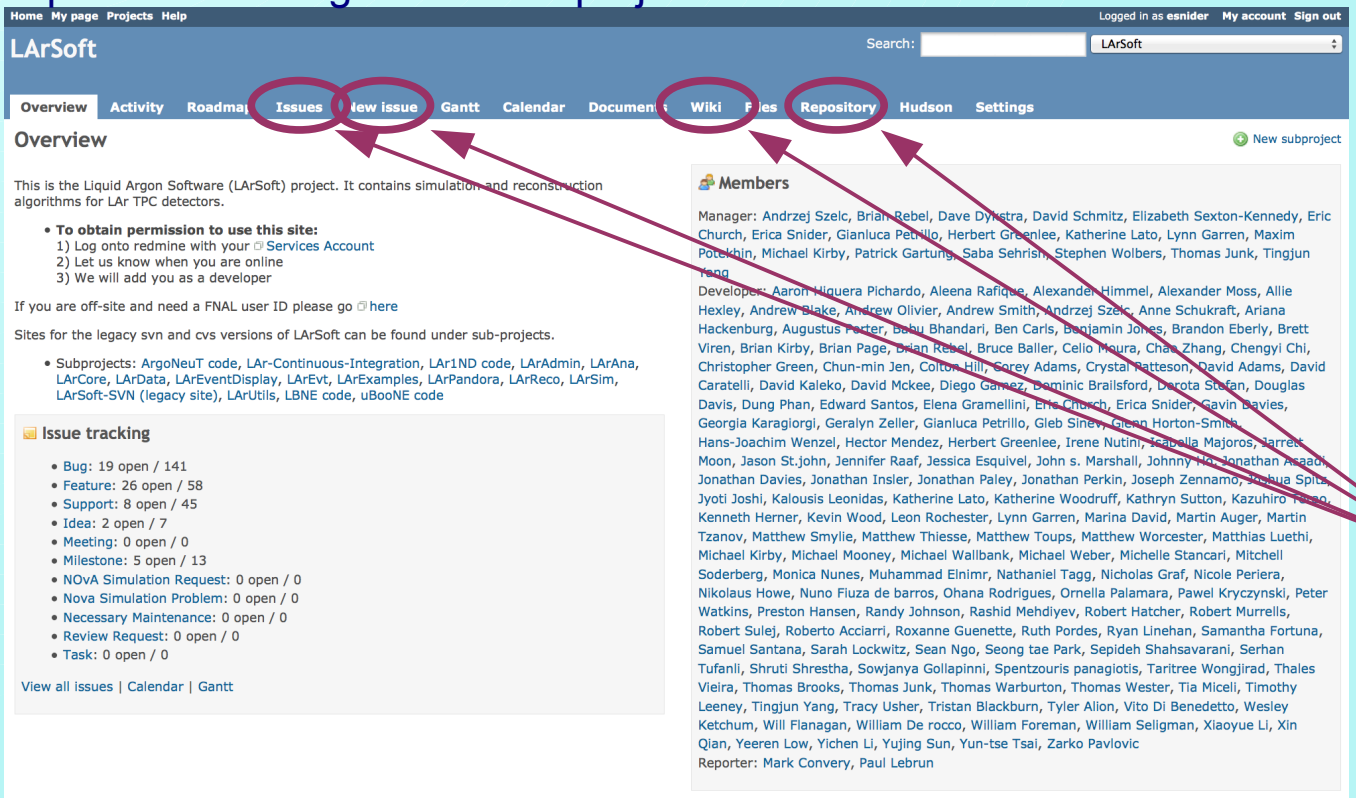
This is the home page for the LArSoft Redmine project

Tabs across the top link to different types of content

LArSoft Redmine site

Redmine sites are called “projects”

https://cdcvs.fnal.gov/redmine/projects/larsoft



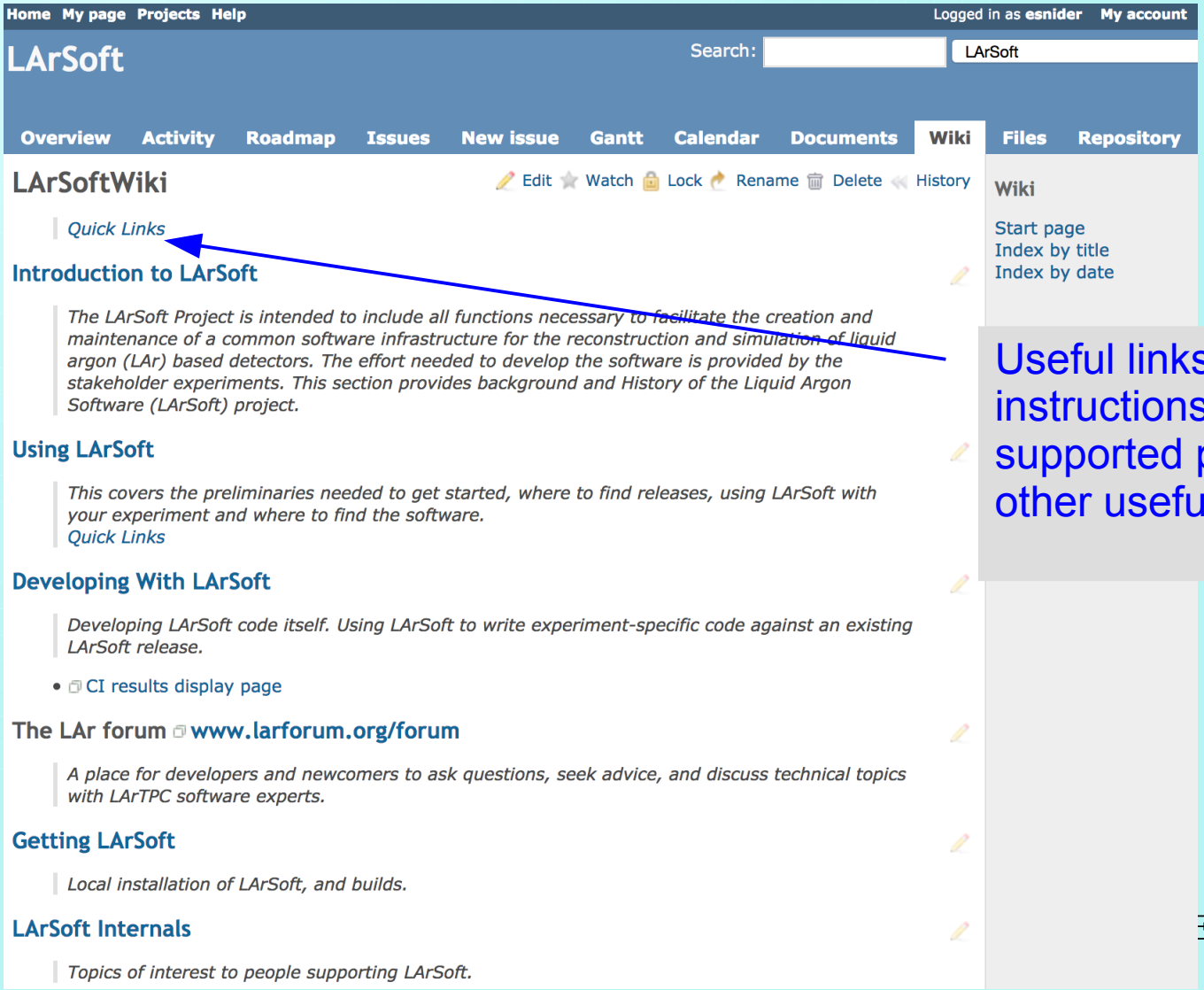
This is the home page for the LArSoft Redmine project

Tabs across the top link to different types of content

The most useful tabs:
- Wiki
- Issues
- New Issues
- Repository

LArSoft wiki

https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki

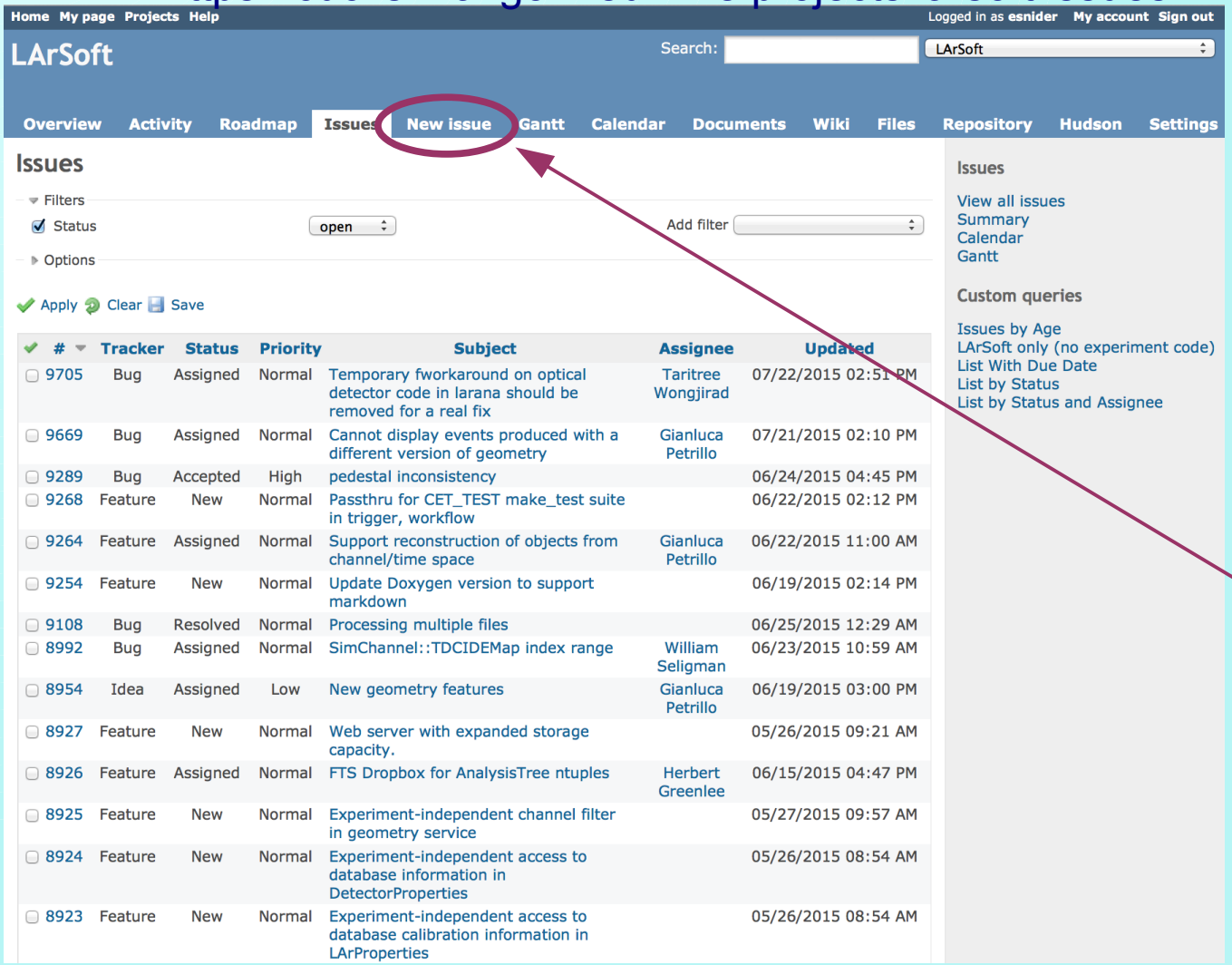


General information and documentation

Useful links to getting started instructions, list of releases, supported platforms and other useful information.

LArSoft issue tracker

https://cdcvs.fnal.gov/redmine/projects/larsoft/issues



We track bugs, problems with LArSoft-related infrastructure, requests for support and new features, questions...

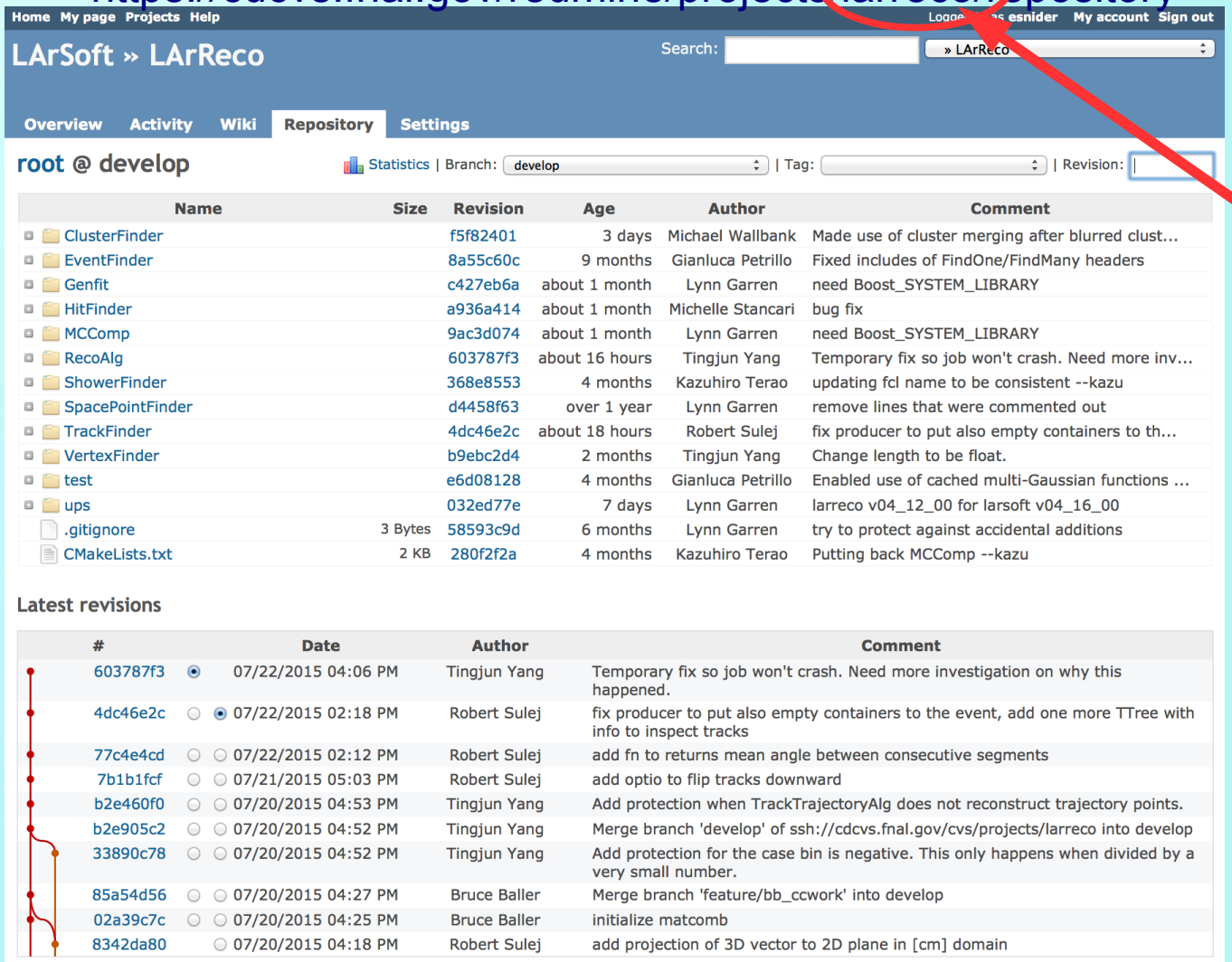
Open a new ticket using the “New Issue” tab if you have any of the above

Create a new issue using this tab.

Must be logged into Redmine using your Fermilab **services account** and password

LArSoft Redmine code browser ...

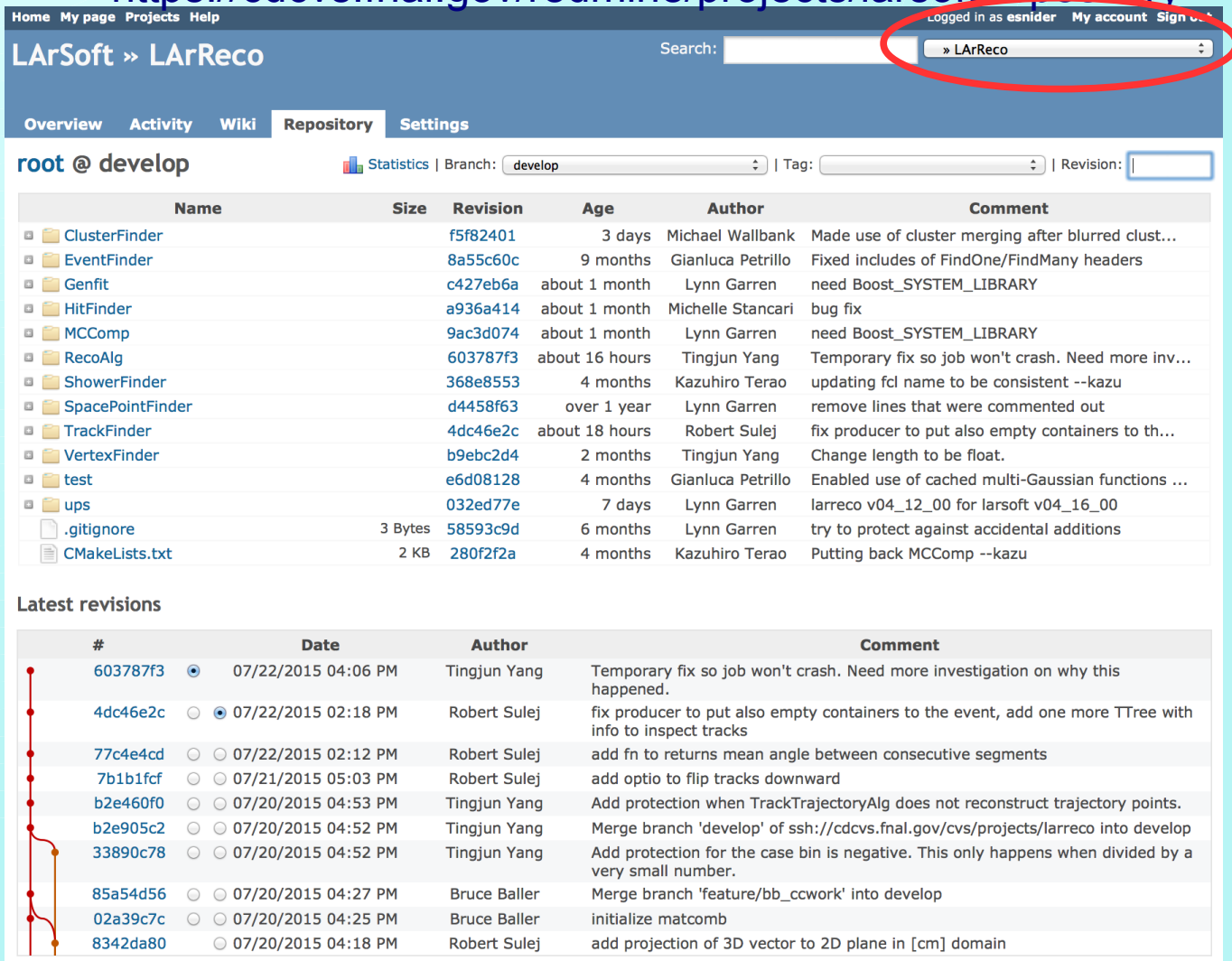
https://cdcvs.fnal.gov/redmine/projects/larreco/repository



Each LArSoft repository lives in a separate Redmine project which is specified here.

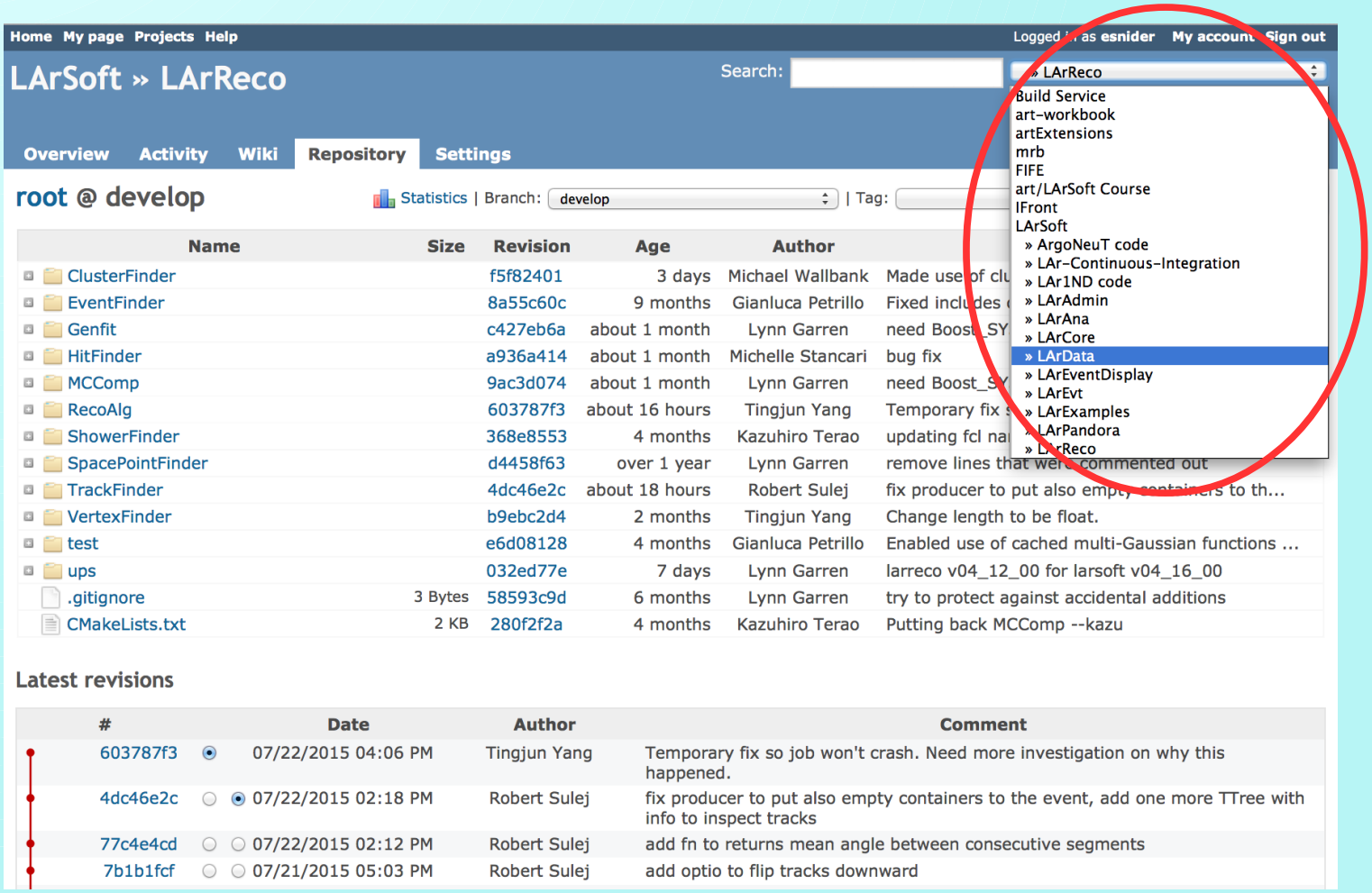
Navigating between LArSoft sub-projects

https://cdcvs.fnal.gov/redmine/projects/larsoft/repository



Can use the project navigation pull-down to get to the desired project.

Navigating between LArSoft sub-projects



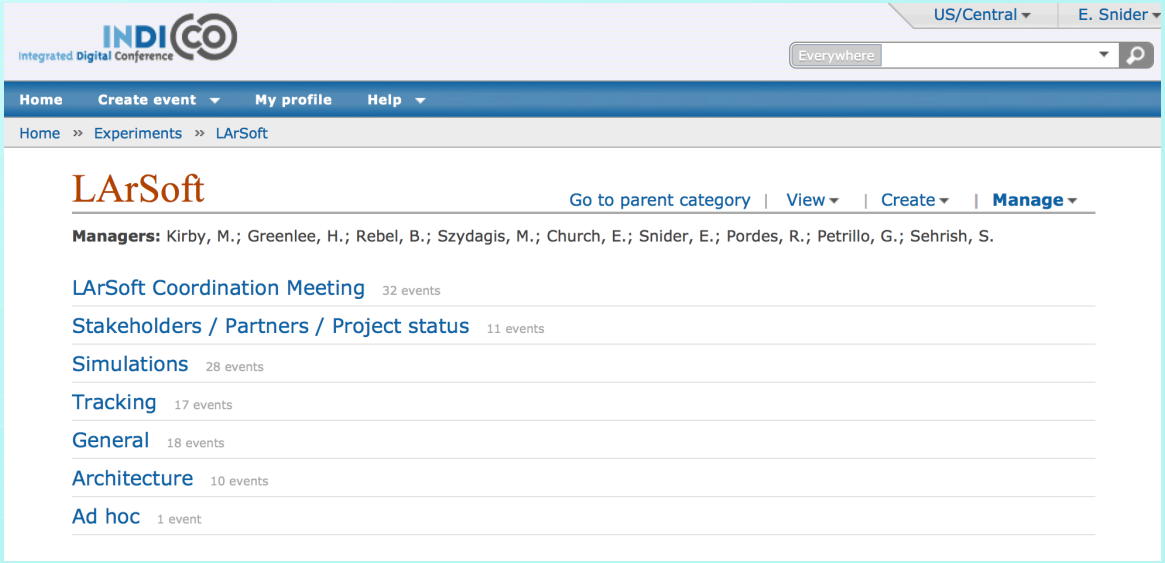
Can use the project navigation pull-down to get to the desired project.

LArSoft Indico site

- Slides and documents from meetings are posted to Indico

  https://indico.fnal.gov/categoryDisplay.py?categId=233

  - Or from the Indico home page: https://indico.fnal.gov/index.py, follow "Experiments", then "LArSoft" links to arrive at the LArSoft page



Can upload slides on the page for the particular meeting...

...but, must be logged in using your **Indico account** and password

Resources

- LArSoft dOxygen documentation system:
  - http://nusoft.fnal.gov/larsoft/doxsvn/html/index.html
- LArSoft email list: larsoft@fnal.gov
  - General announcements. Some technical questions too.
  - Can self-subscribe. See http://listserv.fnal.gov/ for instructions.
- LAr reconstruction software forum: http://www.larforum.org/forum
  - Help with general problems in LAr software
- **LArSoft Coordination Meeting**
  - Bi-weekly at 09:00 Central Time in WH3NE
  - Remote connections via ReadyTalk. Slides posted to LArSoft Indico site.
- LArSoft wiki: https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki
  - Quick page with links to quick-start guides by experiment
- LArSoft issue tracker
  - https://cdcvs.fnal.gov/redmine/projects/larsoft/issues/new
- 2015 LArSoft course material
  - https://indico.fnal.gov/conferenceTimeTable.py?confId=9928#20150807

Core LArSoft support team

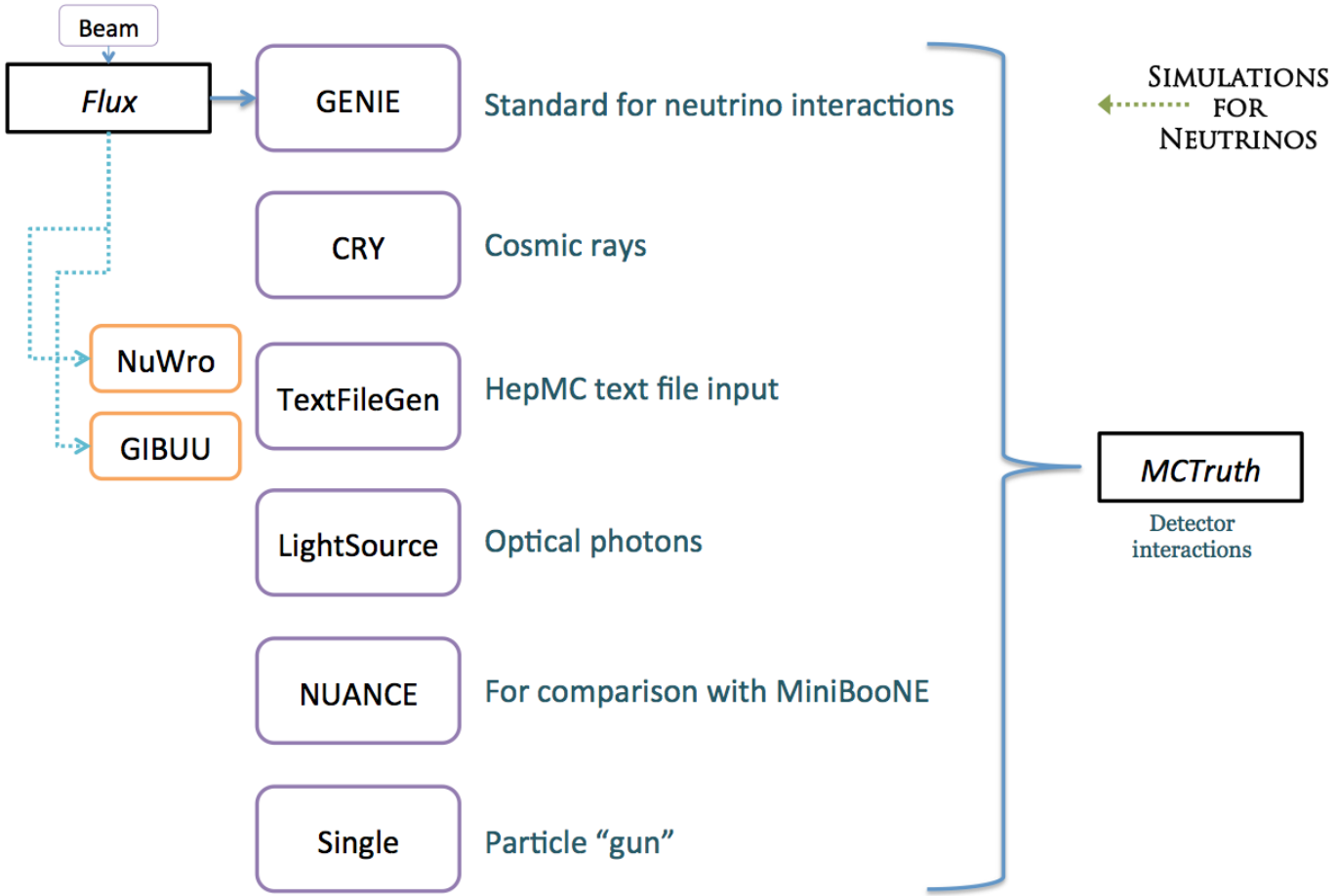
- Core team members
  - Technical lead: Erica Snider
    erica@fnal.gov
  - Project manager: Ruth Pordes (soon to be Katherine Lato)
    ruth@fnal.gov. klato@fnal.gov
  - Lead developer: Gianluca Petrillo
    petrillo@fnal.gov
  - Developer: Saba Sehrish
    ssehrish@fnal.gov
  - Code management and distribution: Lynn Garren
    garren@fnal.gov
  - CI operations and testing support: Vito di Benedetto
    vito@fnal.gov
  - Documentation: Katherine Lato

Email / visit any of the project team!!

The end

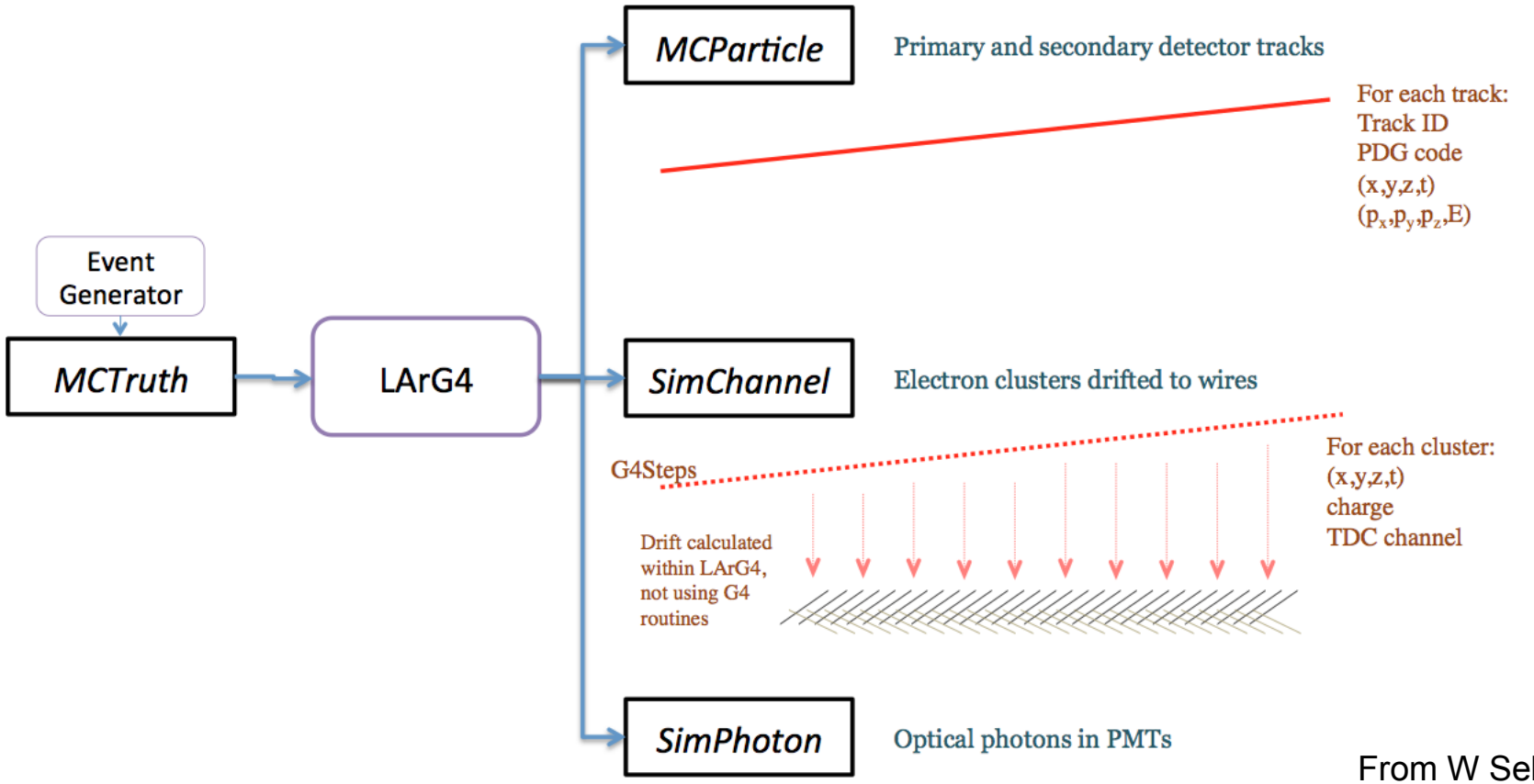
Backup

Event generators



From W Seligman

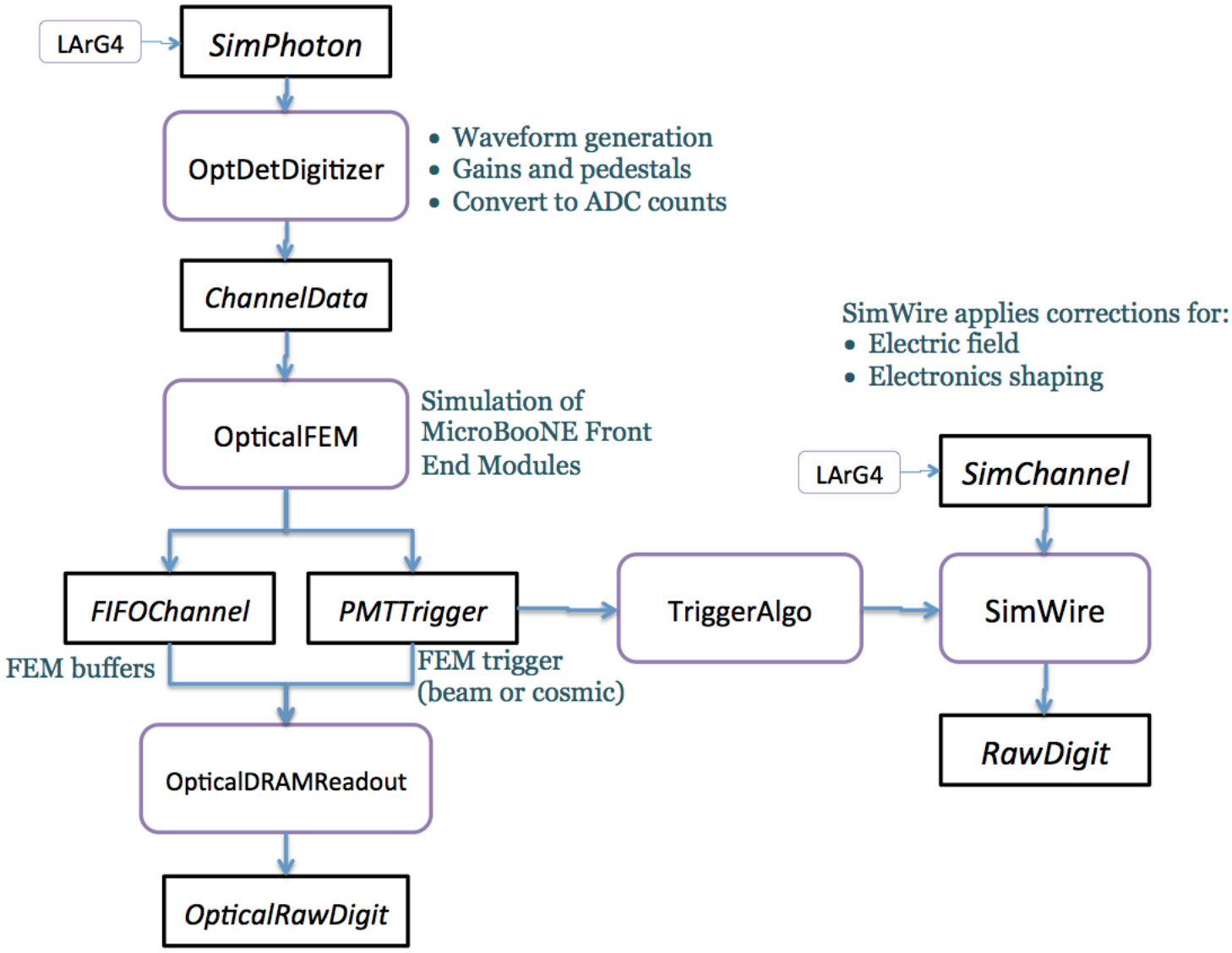
Detector simulation



From W Seligman

Simulation task workflow

Detector response and digitization



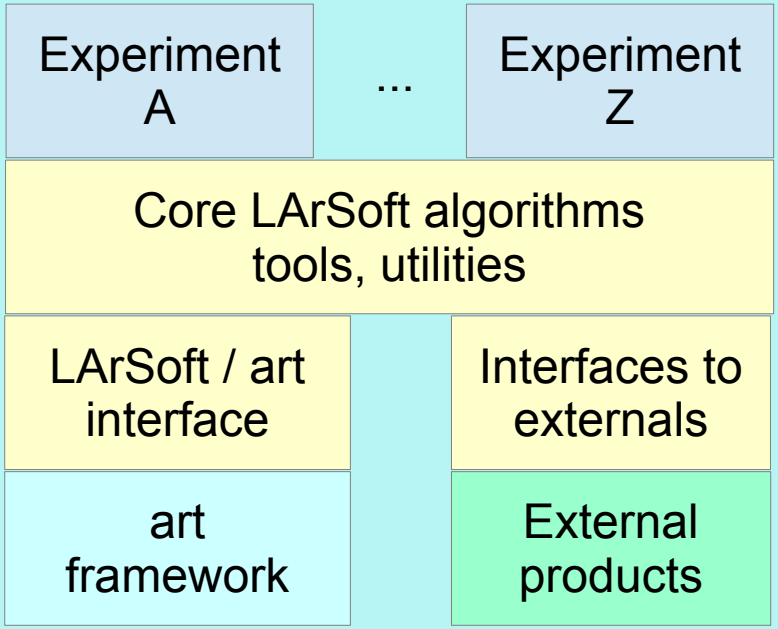
LArSoft design principles and objectives

- Detector interoperability
  - The most important design objective for the LArSoft project
  - Requires care to define (and use!!) common interfaces for accessing detector-specific configuration information
  - Good example: access to detector geometry information
    - A single interface that accommodates different (albeit very similar) geometries
      - Most differences a matter of configuration only
      - Also have detector-specific implementations of the interface where needed
    - Carefully avoid implied geometrical assumptions in algorithms
      - Position of the first plane or wire, the wire spacing, etc.
    - Introduced structures to facilitate generic loops over geometrical elements
      - Define detector / DAQ element IDs at all levels of detector geometry hierarchy
      - Can thereby avoid explicit reference to indices for loops, etc.
  - Also applies to
    - Access to calibration data, LAr properties, detector properties, E-field map, handling of common metadata for data files...
  - Have compiled a long list of do's and don'ts to ensure interoperability

LArSoft design principles and objectives

- Separation of framework and algorithm code
  - Encapsulate algorithms, configuration, tools and utilities into a layer that is independent of the framework
  - Why??
    - Allow testing of small units of algorithm code outside the framework
    - Provide greater flexibility in using algorithms
    - To provide a means of integrating LArSoft code (data products and algorithms) with external frameworks
      - e.g., LArLite used by MicroBooNE for algorithm development, testing

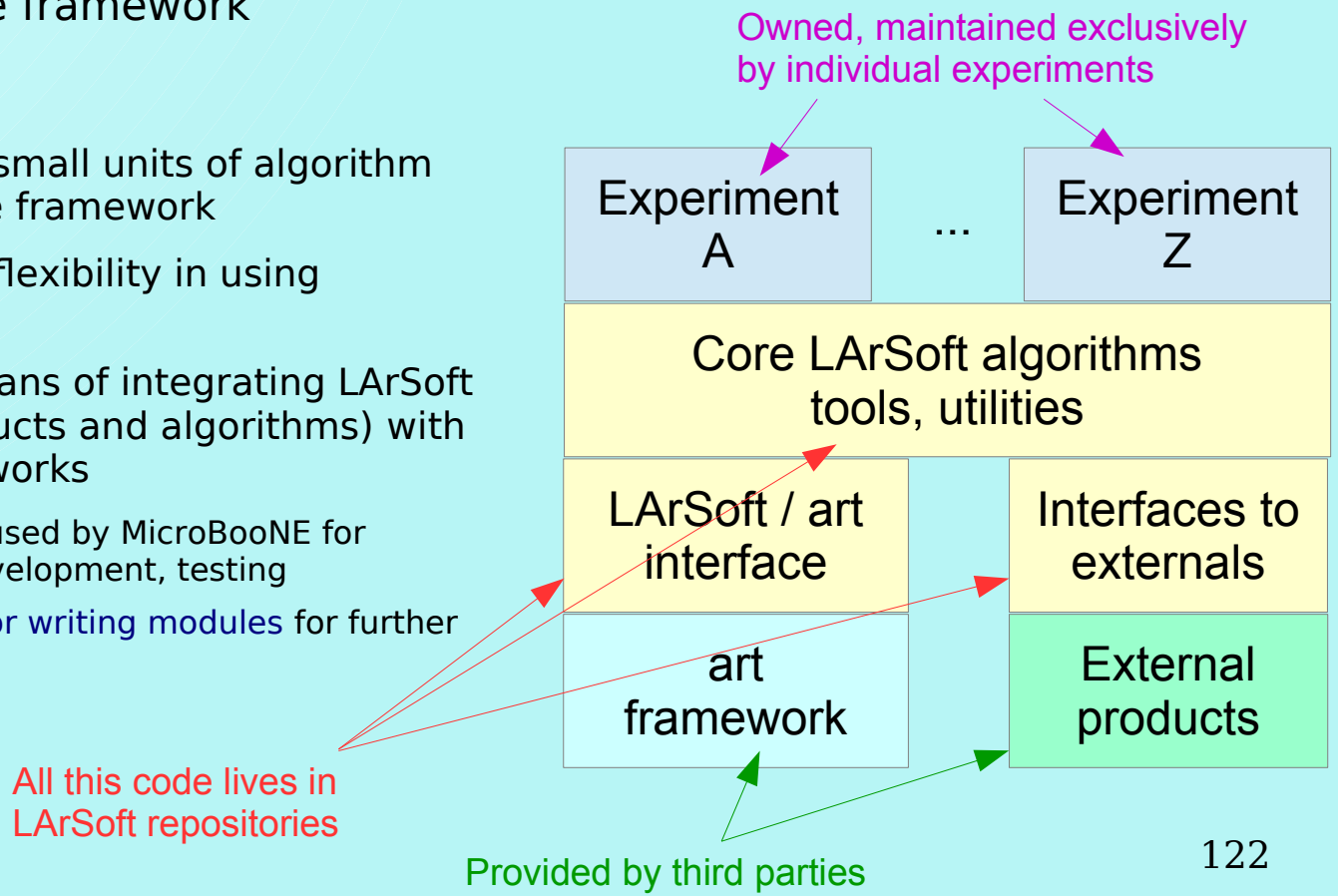
    See art guidance for writing modules for further discussion



LArSoft design principles and objectives

- Separation of framework and algorithm code
  - Encapsulate algorithms, configuration, tools and utilities into a layer that is independent of the framework
  - Why??
    - Allow testing of small units of algorithm code outside the framework
    - Provide greater flexibility in using algorithms
    - To provide a means of integrating LArSoft code (data products and algorithms) with external frameworks
      - e.g., LArLite used by MicroBooNE for algorithm development, testing

    See art guidance for writing modules for further discussion



LArSoft design principles and objectives

- Separation of framework and algorithm code (cont'd)
  - How??
    - Adhere to particular design patterns for the “LArSoft / art interface” code
      - I.e, art modules and services (to be discussed later)
    - Access framework functionality only within the interface code
      - Includes finding input data, writing output data, retrieving any required services, making filter decision calls, etc.
    - Pass all required data, utility classes into algorithms, and all output data back out

General disclaimer: In examining the code, you may note that only a portion of the existing code adheres to this an the other design principles.

- The on-going architecture review project is intended to address this

Strongly encouraging people to adopt this practice for new code.

LArSoft design principles and objectives

- Standardized algorithm interfaces
  - Define standard interfaces for well-defined steps in the workflow so that:
    - Multiple algorithms that address specific problems can share interfaces
  - Promote greater modularity, layering of algorithms
  - Generally enhances flexibility of the code
- Modularity
  - Build sophistication by applying small, targeted algorithms in a layered, iterative structure
  - Leads to code that is more easily tested, more maintainable, more flexible

LArSoft design principles and objectives

- Continuous integration
  - A development scheme in which changes to the main branch of development are integrated and tested frequently
    - At every push to the develop branch
    - Every night
    - At every release

Requires that code authors write tests!

  - Primary LArSoft goals
    - Ensure that code performs as intended
    - Facilitate early detection of problems created in one experiment due to changes introduced by another experiment
    - Ensure that all major features in the develop branch work at all times
  - Are now operating a continuous integration system for LArSoft
    - Currently runs at every push to develop branch
    - Can be triggered manually to run on a non-develop branch of a user's choosing

  See https://cdcvs.fnal.gov/redmine/projects/lar-ci/wiki for details